Contributions to Localization, Mapping and Navigation in Mobile Robotics

Jose Luis Blanco Claraco

13 de Noviembre de 2009

Tesis doctoral en Ingeniería en Telecomunicación

Dpt. de Ingeniería de Sistemas y Automática
Universidad de Málaga

Publicaciones y Divulgación Científica

AUTOR: José Luis Blanco Claraco
EDITA: Publicaciones y Divulgación Científica. Universidad de Málaga

Código ORCID: 0000-0002-9745-285X

UNIVERSIDAD DE MÁLAGA
DEPARTAMENTO DE
DPT. DE INGENIERÍA DE SISTEMAS Y AUTOMÁTICA

El Dr. D. Javier González Jiménez y el Dr. D. Juan Antonio Fernández Madrigal, directores de la tesis titulada "**Contributions to Localization, Mapping and Navigation in Mobile Robotics**" realizada por **D. Jose Luis Blanco Claraco** certifican su idoneidad para la obtención del título de **Doctor en Ingeniería en Telecomunicación**.

Málaga, 13 de Noviembre de 2009

Dr. D. Javier González Jiménez

Dr. D. Juan Antonio Fernández Madrigal

A María

TABLE OF CONTENTS

LIST OF ALGORITHMS

ABSTRACT

This thesis focuses on the problem of enabling mobile robots to autonomously build world models of their environments and to employ them as a reference to self–localization and navigation.

For mobile robots to become truly autonomous and useful, they must be able of *reliably* moving towards the locations required by their tasks. This simple requirement gives raise to countless problems that have populated research in the mobile robotics community for the last two decades. Among these issues, two of the most relevant are: (i) secure autonomous navigation, that is, moving to a target avoiding collisions and (ii) the employment of an adequate world model for robot self-referencing within the environment and also for locating places of interest. The present thesis introduces several contributions to both research fields.

Among the contributions of this thesis we find a novel approach to extend SLAM to large-scale scenarios by means of a seamless integration of geometric and topological map building in a probabilistic framework that estimates the hybrid metric-topological (HMT) state space of the robot path. The proposed framework unifies the research areas of topological mapping, reasoning on topological maps and metric SLAM, providing also a natural integration of SLAM and the "robot awakening" problem.

Other contributions of this thesis cover a wide variety of topics, such as optimal estimation in particle filters, a new probabilistic observation model for laser scanners

based on consensus theory, a novel measure of the uncertainty in grid mapping, an efficient method for range-only SLAM, a grounded method for partitioning large maps into submaps, a multi-hypotheses approach to grid map matching, and a mathematical framework for extending simple obstacle avoidance methods to realistic robots.

ACKNOWLEDGMENTS

In the first place, I must express my gratitude to my advisors Prof. Dr. Javier Gonzalez and Dr. Juan Antonio Fernández Madrigal, who gave me the opportunity to do basic research in the best possible conditions during these last years. I am also indebted to them for the uncountable hours we have spent together discussing about the different research topics and guiding me through the difficult art of writing scientific papers.

Naturally, a research project like this thesis would not have been possible without the great team of people in our lab. Thanks to Dr. Vicente Arévalo and Dr. Cipriano Galindo, who helped me since my first day in the group. I also have good memories from those that were in our lab in the past, specially from Elena Cruz, Jose Luis Reina and of course, Antonio Ortiz de Galisteo *aka* "Tony". A special thank you goes to Javier Cabello, whose huge knowledges about all kinds of mechanic, electric and IT devices had been reflected in unforgettable hours discussing the pros and cons of every possible configuration of our robots – present and future ones. Another great person that has always supported and helped me in these years is Francisco Angel Moreno *aka* "Paco": it is a pleasure to struggle together trying to solve the (probably) unsolvable mathematical problems we always get into. And of course, I also have to mention the rest of PhD students at our lab that are just starting their journey: thank you all for the good moments at work that made it a comfortable place to stay.

I am also grateful to Dr. Paul Newman, who gave me the opportunity of visiting his team, a group of wonderful and talented people. I want to acknowledge all of them here, and also to the rest of research groups I met there: thanks for the warm welcome and for the funny moments during my visit and whenever we have met later.

Also worthy remembering, all those people whom I met at the different conferences

and events around the world: thanks for making each trip an enriching experience.

Thanks to the thesis commitee members, Achim Lilienthal, Alfonso García, José Neira, Juan Andrade and Simon Lacroix, whose insighful comments and recommendations on how to improve the text have been taken into account in this corrected version. Furthermore, I want to mention my two external reviewers for this PhD thesis: Dr. Ingmar Posner and Dr. Patrick Pfaff. Thank you for scheduling such a tough work among so many other deadlines.

Finally, I feel extremely grateful to my family, which has always supported me not only materially but more importantly, believing in me and inspiring me throughout this research career. Last but not least, I am completely indebted to María, who has not only suffered my uncountable working hours at nights or holidays, but also always encouraged me in the hardest moments. Yours is the warmest thank you.

CHAPTER 1

INTRODUCTION

1.1 Towards autonomous robots

By the end of 2007 there were about 1 million functional robots in the world, virtually all of them industrial arm robots [IFR08]. In comparison to the evident success of robotics in industrial scenarios, *mobile robots* still have a marginal relevance in our society, although exceptions can be found as the dozens of automatic cleaning machines that operate daily at the Paris metro [Kuh97] or the millions of cleaning robots sold by iRobot for personal use in homes [iRo].

Nowadays, AGVs (autonomous guided vehicles) usually operate at facilities purposely modified with guidewires or other devices [Hah] that define the possible trajectories the robot can follow. Extending the usage of robots to homes, hospitals and public places in general represents an extremely challenging step that state-of-the-art research has not fully solved yet. Achieving such a level of integration of mobile robots in our society demands addressing a number of issues in order for them to become fully *autonomous*. To only mention a few of these hurdles, an autonomous robot [Sie04]

that interacts with humans should have the abilities of grasping and manipulating objects [Shi96], correctly managing social interactions and understanding non-verbal language [Fon03], recognizing people [Zha03] and planning complex tasks [Gal04], among others. Naturally, the list of required abilities may vary depending on the domain and purpose of the robot. For instance, aerial or underwater [Yuh00] autonomous robots must be adapted differently than those created to assist the elder people or for entertainment in museums [Bur99].

Nevertheless, there exist two basic issues that any mobile robot must solve in order to achieve a certain degree of autonomy, namely being able to track its own position within the world (*localization*) and driving itself in a safely manner towards some given target location (*navigation*). These two topics are, broadly speaking, the concern of the present work.

Regarding mobile robot localization, the issues to be faced greatly depend on low-level issues, such as the kind of employed sensors (e.g. cameras, laser range finders or radio beacons). Nevertheless, it must be remarked that despite those differences, a probabilistic treatment of the problem has revealed itself as a fruitful approach, particularly since the introduction of particle filtering techniques in the late 1990s [Del99, Thr01b]. In the common situation addressed in robot localization, the initial position of the vehicle is only roughly known (or totally unknown for the so called "awakening problem" [Fox99a]), and the robot must gain increasing confidence about its location as it moves around and gathers sensory data. This process is possible due to the existence of data previously known by the robot, usually a *map* of the environment entered by the human.

A natural evolution of this problem is to go one step further and let the robot start without any previous knowledge about its environment, that is, without an already

built map. In this case the robot must localize and build a map as it senses the environment for the first time. This paradigm, called the Simultaneous Localization and Mapping (SLAM) problem, has witnessed a highly intense research activity during the last two decades [Bai06a, Thr02, Thr05]. The special interest of the robotics community in SLAM can be easily understood by realizing its potential benefits: a mobile robot capable of *learning* the characteristics of its environment would not need any special setup or modification in its workplace.

As with the localization problem, probabilistic approaches to SLAM are specially appropriate due to their capability to deal with noise in the sensors and with uncertainty in both the position of the robot and the world model, which will be always uncertain due to the noisy and sometimes ambiguous nature of robotic sensors and actuators.

An interesting reflection is the fact that both localization and SLAM are just special cases of one single statistical problem, namely *probabilistic estimation* [Bis06]. However, there are several different ways of approaching these problems, some more appropriate than others depending on the kind of robotic sensors or map representation employed. This is one of the reasons of the huge body of existing literature in the field in spite of all the works reducing to the same basic mathematical problem.

Last but not least, autonomous mobile robots should navigate safely through their environment, which involves the usage of techniques aimed at finding feasible paths for the robot to reach the desired targets while avoiding collisions with static or dynamic obstacles. These approaches strongly depend on the mechanical locomotion system of the robot. Although there is an increasing trend to design biped robots [Goe07, Par01], this thesis focuses on the more common case of wheeled robots. Motion planning and control for these robots is a difficult problem due to non-holonomic restrictions that limit the set of feasible maneuvers.

1.2 Contributions of this thesis

The contributions of this thesis are all advanced solutions to the above-mentioned requirements for autonomy, that is, localization, mapping and navigation. A summary of the most relevant contributions follows below:

- A novel Bayesian approach for the problems of localization and SLAM based on probabilistic estimation over a hybrid metric-topological (HMT) state-space. This new approach, called HMT-SLAM, allows robust map building of large-scale environments and also provides an elegant integration of the "robot awakening" and local SLAM problems. The publications [Bla07b] and [Bla08d] are the most relevant results related to this line of research.

- A new particle filter algorithm for optimal estimation in problems with non-parametric system and observation models. This algorithm has direct applications to both localization and SLAM, and it was published in [Bla08h].

- The introduction of a new stochastic sensor model (a *likelihood model* in Bayesian estimation) for precise range finders based on Consensus Theory, which is shown to perform better than previous methods for dynamic environments. The method was reported in [Bla07c].

- A probabilistic method aimed at improving the accuracy of the motion model of mobile robots by means of fusing readings from odometry and a gyroscope, reported in [Bla07d].

- A new measurement of uncertainty for global map-building that unifies uncertainty in the estimates of the robot path and the map and provides a more

valuable information than previous proposals in the context of autonomous exploration, as reported in [Bla06a, Bla08c].

- Probabilistic solutions to SLAM with range-only sensors, which led to [Bla08e, Bla08a]. These works are in fact the most recent ones from a series of works on localization with Ultra-Wide-Band (UWB) radio beacons ([FM07, Gon07, Gon09a]).

- The proposal of a method for partitioning observations into approximately independent clusters (*local maps*), a necessary operation in HMT-SLAM. This new approach was introduced in [Bla06b, Bla09b].

- Matching occupancy grid maps, another operation needed in HMT-SLAM. The results of this research line appeared in [Bla07e, Bla10b, Bla10c].

- A new approach to mobile robot obstacle avoidance through multiple space transformations that extend the applicability of existing obstacle avoidance methods to kinematically-constrained vehicles, making no approximations in either the robot shape or in the kinematic restrictions. A series of works in this line has previously appeared in [Bla06c, Bla08f, Bla08g].

1.3 Framework of this thesis

This thesis is the outcome of five years of research activity of his author as a member of the MAPIR group [1], which belongs to the *Departamento de Ingeniería de Sistemas y Automática* [2] of the University of Málaga. During that time, funding was supplied by the Spanish national research contract DPI2005-01391, the Spanish FPU grant program and the European contract CRAFT-COOP-CT-2005-017668. The scientific production of this period is in part reflected in this thesis, while the results that had direct applications to the industry have led to three patents [Est09, Jim07, Jim06].

The MAPIR group experience in mobile robotics goes back to the early 90s, with several PhD theses in the area, such as [GJ93] and [RT01]. Moreover, two other PhD theses [FM00, Gal06] addressed the topic of large-scale graph world models, which is closely related to HMT-SLAM, one of the main contributions of this thesis.

The author completed the doctoral courses entitled "Ingeniería de los Sistemas de Producción" (*Production System Engineering*) given by the *Departamento de Ingeniería de Sistemas y Automática*, and also complemented his academic education with the participation in the *SLAM Summer School* (2006), and a three months stay with the *Mobile Robot Group* headed by Dr. Paul Newman.

Regarding the experimental setups presented in this work, they have been carried out with two mobile robots: SENA, a robotic wheelchair [Gon06a], and SANCHO, a fair-host robot [Gon09b]. The author, among the rest of the group, has devoted great efforts to the development of these robots, including both software implementation tasks and the design of electronic boards.

[1] http://babel.isa.uma.es/mapir/
[2] http://www.isa.uma.es/

1.4 Organization

After this introduction, Chapter 2 provides the fundamental background on the probabilistic techniques that will be employed throughout the rest of the thesis.

The work is divided into four clearly-differentiated parts:

- Chapters 3 to 6 are devoted to purely metric robot localization techniques.
- Chapters 7 to 10 cover the contributions related to mobile robot autonomous exploration and SLAM (also in a purely metric setting), including a review of existing approaches.
- Chapters 11 to 14 present the new approach to local and topological localization and map building, called HMT-SLAM. Individual operations within this framework, such as map partitioning or grid map matching, are discussed in their corresponding chapters.
- Finally, chapters 15 to 16 review existing local navigation approaches and introduce a new space transformation for efficient search of collision-free trajectories, with applications to motion planning and reactive navigation.

CHAPTER 2

PROBABILISTIC FOUNDATIONS

> "All events, who by their smallness seem not consequences of natural laws, are in fact outcomes as necessary as celestial movements [...] but because of our ignorance of the immense majority of the data required and our inability to submit to computation most of the data we do know of, [...] we attribute these phenomena to randomness, [...] an expression of nothing more than our ignorance."
>
> Théorie Analytique des Probabilités (1812), pp. 177-178
> Pierre-Simon Laplace

2.1 Introduction

Probability theory has become an essential tool in modern engineering and science. The reason of this success can be found in the unpredictable aspects of any system under real-world conditions: independently of the precision of the employed instruments, measurements are always noisy and thus will vary from one instant to the next. Another

difficulty of practical systems is that we are often interested in measuring the state of systems not directly *observable*, where the only measurable information is non-trivially related to that state.

A probabilistic treatment of such situations has the potential to provide us with explicit and accurate estimates of the system, right from the accumulated knowledge gained through, possibly noisy and indirect, observations.

With the intention of keeping this thesis as much self-contained as possible, subsequent sections introduce fundamental concepts and mathematical definitions on which probabilistic mobile robotics relies. Nevertheless, this chapter does not intend to be a thorough and complete reference on probability theory, for which the interested reader is referred to the excellent existing literature [Apo07,Bis06,Dou01,Ris04,Rus95a, Thr05].

2.2 Basic definitions

Let S be the *countable* (finite or infinite) set of all the possible outcomes of an experiment. Then, a *probability space* is defined as the triplet $(S, \mathcal{B}, P)$ with $\mathcal{B}$ being a Boolean algebra (typically the set of all the possible subsets of S) and with P being a *probability measure*, which provides us the *probability* of every possible *event* (the combination of one or more outcomes from S). The probability measure P must fulfill a few fundamental requirements, namely, (i) providing a total probability of 1 for the entire set S, (ii) being nonnegative and (iii) being *completely additive*, which can be informally defined as any function whose output for an input set $A = \{a_1, ..., a_i\}$ is the sum of the outputs for each individual element in the set [Apo07].

In the case of *uncountable* sample spaces S (e.g. the line of the real numbers $\mathbb{R}$), the smallest applicable Boolean algebra is a kind of σ-algebra called *Borel algebra*, built

from all the intervals where S is defined. The resulting measures (or *probabilities*) emerging from such algebra are *Lebesgue measurable*, which means that probability distributions (to be defined shortly below) may contain any *countable* number of discontinuities as long as all the discontinuities belong to a *null set* [Wei73].

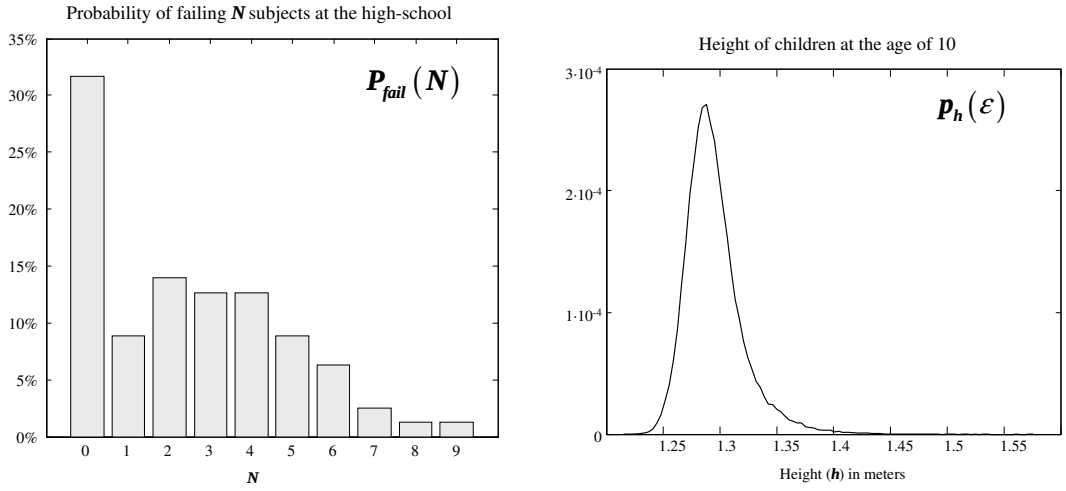
Following the common practice in science and engineering texts, the rest of this thesis employs the most natural Boolean algebra $\mathcal{B}$ for *probability spaces*: a Borel σ-algebra for the common case of S being a subset of $\mathbb{R}^n$, and the set of all possible subsets for S being a discrete sample space. Therefore, these technical details about Measure Theory will rarely be mentioned from now on. For further discussion about this *modern* description of Probability Theory the reader is referred to the existing literature [Apo07].

In the usual approach to a probability theory problem we will be interested in *random variables* (RVs). A RV represents any measurable outcome of an experiment, thus each realization of the variable or *observation*, will vary for each repetition of the experiment (e.g. the time to finish a race). RVs may also represent non-observable quantities (e.g. the temperature at the core of the sun) of which indirect measurements exist.

A RV is properly modeled by its *probability distribution* [1], the function that determines how *likely* is to obtain one or another outcome for each observation of the variable.

In the case of a *discrete* random variable X, its distribution is defined by means of a *probability mass function* (pmf), written down as $P_X(X_i)$ or $P(X = X_i)$ (note the capital P), and which reflects the probability of X to be exactly X_i, for the whole domain of possible values of X_i.

[1] Note that this corresponds to the additive set function P in the probability space triplet defined above.



(a) (b)

Figure 2.1: Examples of probability distributions for (a) a discrete variable (pmf) and (b) a continuous variable (pdf).

For a *continuous* variable x, the distribution is described by its *probability density function* (pdf), denoted by $p_x(\epsilon)$ (note the uncapitalized p) which does not indicate probabilities, but *probability densities* for each possible value of $x = \epsilon$. Here, probabilities can be always obtained for the event that the variable x falls within a given range $[a, b]$ by means of:

$$P(a < x < b) = \int_a^b p_x(\epsilon) d\epsilon \tag{2.2.1}$$

The concepts of pmf and pdf are illustrated with two realistic examples in the Figure 2.1.

Another concept, related to the pdf through Lebesgue integration, is the *cumulative distribution function* (cdf), which also fully describes a probability distribution by measuring the probability of the variable x being *below* any given value. In the case of a real-valued variable x the cdf is represented by F_x, such as

$$F_x(\epsilon) = P(x < \epsilon) \tag{2.2.2}$$

Obviously, the limit of a cdf must be zero and one for x tending towards minus and plus infinity, respectively. A cdf may contain discontinuities, but continuity from the right hand must be assessed for each of them [Apo07]. For the common case of a *continuous* cdf, the probability of any given *open* set $]a, b[$ will be always identical to that of the *closed* set $[a, b]$, since the Lebesgue measure of the end points a and b is zero.

A distribution is said to be *multivariate* when it describes the statistical occurrence of more than one variables, which may be continuous, discrete or an arbitrary mixture of them. The pdf (or cdf) of multiple variables is called a *joint* distribution, in contrast to a *marginal* distribution where only one of the variables is considered.

Another important concept is the *mathematical expectation* of any arbitrary function $f(x)$ of a RV x, which determines the output from the function $f(\cdot)$ that will be closest, in average, to the output generated from random samples of x. It is denoted with the $E[\cdot]$ operator, defined as

$$E_x[f(\epsilon)] = \int_{-\infty}^{\infty} f(\epsilon) p_x(\epsilon) d\epsilon \tag{2.2.3}$$

Using this expectation we can define the n'th (non-central) moment of a distribution as:

$$\mu'_n = E_x[\epsilon^n] \tag{2.2.4}$$

The first of these moments ($n = 1$) is the *mean* of the distribution. Denoting the mean of the RV x as $\bar{x}$ we can now determine the n'th *central moment* of its distribution, given by:

$$\mu_n = E_x[(\epsilon - \bar{x})^n] \tag{2.2.5}$$

In this case, the most widely employed moment is the second order one, named *variance*, which measures the dispersion of the distribution. In the case of multivariate distributions the moments become matrices, and the second central moment is called the *covariance matrix*.

Regarding the probability distributions of two or more RVs, there exist a number of fundamental definitions that will be used extensively throughout this thesis. They are enumerated briefly below.

Conditional distribution: The conditional pdf of a RV y *given* another variable x is written down as $p(y|x)$, and can be shown to be:

$$p(y|x) \doteq \frac{p(x,y)}{p(x)} \tag{2.2.6}$$

with $p(x,y)$ being the joint pdf of both variables.

Independence: A pair of variables x and y are said to be *independent* if the information provide bdy one of them contributes nothing to the knowledge about the other. Put mathematically, independent RVs fulfill:

$$p(y|x) = p(y) \tag{2.2.7}$$

$$p(x|y) = p(x) \tag{2.2.8}$$

$$p(x,y) = p(x)p(y) \qquad \text{(A consequence of Eqs. (2.2.6)–(2.2.7))}$$

Conditional independence: A pair of variables x and y are conditionally independent given another third variable z if, given knowledge of z, further knowledge of x or y gives no information about y or x, respectively. This property will be discussed shortly in §2.10.

Law of total probability: The pdf of a set of one or more RVs y can be always obtained by "averaging out" its conditional distribution $p(y|x)$ for any other arbitrary set of RVs x, that is:

$$p(y) = E_x[y|x] = \int_{-\infty}^{\infty} p(y|x)p(x)dx \tag{2.2.9}$$

In the case of discrete variables, this law can be expressed as a sum:

$$P(y) = E_x[y|x] = \sum_{\forall x} P(y|x)P(x) \tag{2.2.10}$$

2.3 Parametric probability distributions

Probability distributions of random variables can be broadly divided into two distinctive groups: *parametric* and *non-parametric*.

In the first case, the pmf or pdf governing the random variable can be described through a mathematical equation that gives the exact value of the distribution for the whole domain of the variable. In contrast, non-parametric distributions cannot be modeled by formulas that yield directly and exactly their value, and other methods must be employed instead as will be discussed below (§2.4).

There exist dozens of well-known parametric distributions [Abr65, Eva01], although in this work only a few will be employed, which are introduced below.

2.3.1 Uniform distribution

This is the simplest distribution, where all the potential values of a random variable z with a domain $[a, b]$ have exactly the same occurrence likelihood:

$$z \sim \mathcal{U}(a,b) \qquad p_z(x) = \mathcal{U}(x;a,b) = \begin{cases} \frac{1}{b-a} & \text{, for } x \in [a,b] \\ 0 & \text{, otherwise} \end{cases} \tag{2.3.1}$$

2.3.2 Normal or Gaussian distribution

Certainly the most widely employed density due to its remarkable properties, such as the convenience of many operations such as summing, conditioning or factoring normal variables also giving normal variables.

In this text, the fact that an N-dimensional random variable $\mathbf{z}$ with mean $\bar{\mathbf{z}}$ and covariance matrix $\mathbf{\Sigma_z}$ is governed by a Gaussian distribution will be denoted as:

$$\mathbf{z} \sim \mathcal{N}(\bar{\mathbf{z}}, \mathbf{\Sigma}_z) \tag{2.3.2}$$

with its pdf defined parametrically as:

$$p_{\mathbf{z}}(\mathbf{x}) = \mathcal{N}(x; \bar{\mathbf{z}}, \mathbf{\Sigma}_z) = \frac{1}{(2\pi)^{N/2}\sqrt{|\mathbf{\Sigma_z}|}} \exp\left(-\frac{1}{2}(\mathbf{x}-\bar{\mathbf{z}})^\top \mathbf{\Sigma_z}^{-1}(\mathbf{x}-\bar{\mathbf{z}})\right)$$

2.3.3 Chi-square distribution

Given a sequence of k uni-dimensional normally-distributed variables z_i with mean zero and unit variance, the statistic

$$q = \sum_{i=1}^{k} z_i^2 \tag{2.3.3}$$

is also a random variable that follows a chi-square distribution with k degrees of freedom, that is, $q \sim \chi_k^2$. The density of this distribution, defined for non-negative numbers only, is given by:

$$p_q(x) = \frac{x^{k/2-1}e^{-x/2}}{2^{k/2}\Gamma(k/2)} \tag{2.3.4}$$

where $\Gamma(k)$ is the gamma function, with no closed-form solution.

The importance of this density comes from it being the foundation of an important method to compare two statistical distributions, named the *chi-square test*. This test is widely employed in probabilistic robotics to deal with problems such as stochastic data association (see [Nei01] and §14.6). In a chi-square test we are not directly interested in the chi-square pdf, but rather on its *inverse cumulative distribution function*, which will be denoted as $\chi^2_{k,c}$. The test gives us the maximum value of the statistic q such as it can be asserted that the two distributions being compared coincide with a certainty of c (a probability between 0 and 1).

2.4 Non-parametric probability distributions

In spite of the wide applicability of the normal distribution, many continuous variables in the real-world cannot be properly modeled as Gaussians or any other parametric distribution. In those cases, non-parametric methods must be applied to model the corresponding probability densities.

Kernel-based methods approximate the actual density distribution of a random variable from a sequence of *independent* samples. A kernel function, usually a Gaussian [Par62], is inserted at each sampled value, and the pdf estimation consists of the average of all these kernels. Since in mobile robotics there are few situations ([Lil07] is an example) where we can draw a large number of independent samples from the variables of interest, these methods have a modest presence in the field.

The most straightforward non-parametric estimators are *histograms* (for 1-dimension) or *probability grids* (for higher dimensions). Although such approximations did find their application to mobile robot localization [Fox99b], it quickly became evident that

keeping the probability for each *bin* in the histogram (or *cell* in the grids) is impractical and wastes most of the computational and storage resources in, probably, non-interesting parts of that space.

An alternative approach is *importance sampling* [Dou01], where the actual pdf is approximated by a set of weighted samples (or *particles*) which are usually concentrated on the relevant areas of the state space. The validity of importance sampling is based on the interesting property that any statistic $E_x[f(x)]$ of a random variable x can be estimated from a set of M samples $x^{[i]}$ with associated weights $\omega^{[i]}$, since it can be proven that

$$\lim_{M \longrightarrow \infty} \sum_{i=1}^{M} \omega^{[i]} f(x^{[i]}) = E_x[f(x)] \tag{2.4.1}$$

for the correct values of weights. In practice, the degree of accuracy attainable by importance sampling is an important issue, since the number of samples must be kept bounded due to computational limitations. Montecarlo simulation with importance sampling is the base of particle filters, which will be discussed in §2.7.2.

2.5 Mahalanobis distance

The Mahalanobis distance D_M is a convenient measurement of distances from a fixed point $\mathbf{y}$ to another point $\mathbf{x}$ whose location follows a normal distribution $\mathcal{N}(\bar{\mathbf{x}}, \mathbf{\Sigma_x})$. For two independent variables, this distance is given by:

$$D_M(\mathbf{x}, \mathbf{y}) = \sqrt{(\mathbf{y} - \bar{\mathbf{x}})^\top \mathbf{\Sigma_x}^{-1} (\mathbf{y} - \bar{\mathbf{x}})} \tag{2.5.1}$$

The intuitive idea behind the Mahalanobis distance is to account for distances weighted by their uncertainty in each dimension. Figure 2.2 illustrates this with ellipses

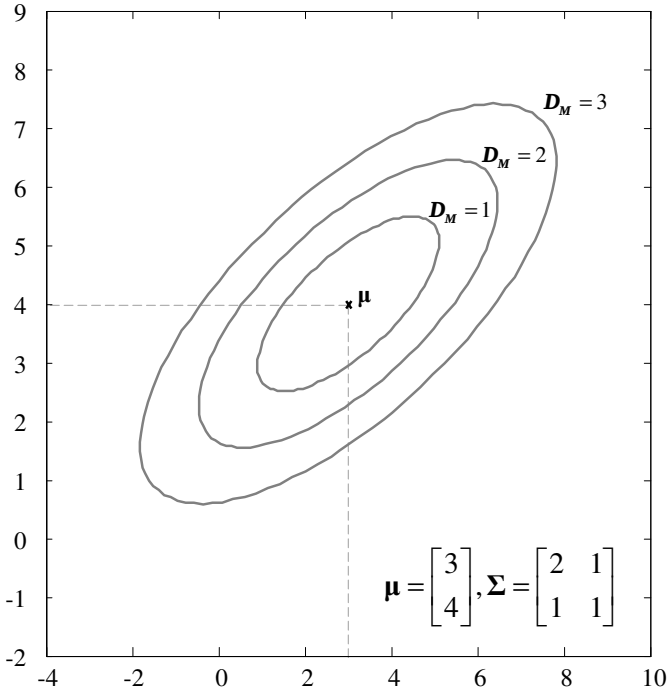


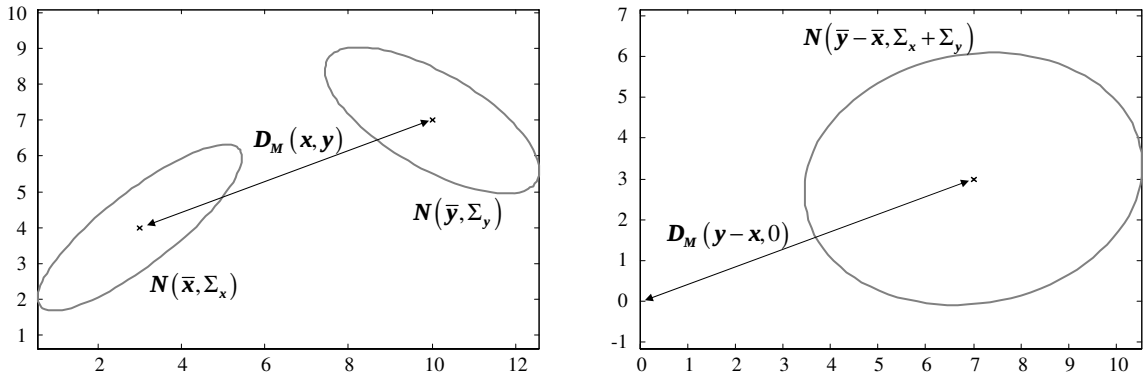
Figure 2.2: An example of the Mahalanobis distance for a 2-d Gaussian $\mathcal{N}(\mu, \boldsymbol{\Sigma})$. The ellipses represent the 68%, 95% and 99.7% confidence intervals for the random variable described by this Gaussian to be within the enclosed areas, for D_M being 1, 2 or 3, respectively.

of constant Mahalanobis distance from the mean point of a 2-d Gaussian. These ellipses will be often used to represent *confidence intervals* throughout this text.

Another interesting application of the Mahalanobis distance is when both points x and y are Gaussian random variables. In that case, the Mahalanobis distance can be computed as:

$$D_M(\mathbf{x}, \mathbf{y}) = \sqrt{(\bar{\mathbf{y}} - \bar{\mathbf{x}})^\top (\boldsymbol{\Sigma}_\mathbf{x} + \boldsymbol{\Sigma}_\mathbf{y} - 2\boldsymbol{\Sigma}_{\mathbf{xy}})^{-1} (\bar{\mathbf{y}} - \bar{\mathbf{x}})} \tag{2.5.2}$$

with $\boldsymbol{\Sigma}_{\mathbf{xy}}$ being the cross-covariance matrix of the two variables. This expression can be easily derived by noticing that computing this distance between two Gaussians can be reformulated as the distance from the origin to a new random variable $\delta = \mathbf{y} - \mathbf{x}$, which follows the distribution $\mathcal{N}(\bar{\mathbf{y}} - \bar{\mathbf{x}}, \boldsymbol{\Sigma}_\mathbf{x} + \boldsymbol{\Sigma}_\mathbf{y})$ as illustrated in Figure 2.3.



(a) (b)

Figure 2.3: (a) An example of the Mahalanobis distance between two Gaussians x and y. (b) The same distance, reformulated as the distance to the origin from the variable representing the difference of both Gaussians.

2.6 Kullback-Leibler divergence

The Kullback-Leibler divergence (KLD) is a measure of the similarity between two probability density functions $p(x)$ and $q(x)$, defined as [Kul51]:

$$D_{KL}(p,q) = \int_{-\infty}^{\infty} p(x) \log \frac{p(x)}{q(x)} dx \tag{2.6.1}$$

Only two exactly equal densities give a KLD value of zero. Otherwise, the KLD is always a positive quantity. In spite of the similarities with the mathematical definition of a *distance*, the KLD is not a valid distance metric since, in general, $D_{KL}(p,q) \neq D_{KL}(q,p)$, hence the usage of the term *divergence*.

One application of KLD found in mobile robotics is to measure the similarity of two independent N-dimensional Gaussian distributions p and q, that is, $D_{\mathrm{KL}}(p,q)$, provided that:

$$\begin{aligned} p(\mathbf{x}) &= \mathcal{N}(\mathbf{x}; \mu_p, \mathbf{\Sigma}_p) \\ q(\mathbf{x}) &= \mathcal{N}(\mathbf{x}; \mu_q, \mathbf{\Sigma}_q) \end{aligned} \tag{2.6.2}$$

which has the closed-form solution:

$$D_{\mathrm{KL}}(p, q) = \frac{1}{2}\left(\log\frac{|\mathbf{\Sigma}_q|}{|\mathbf{\Sigma}_p|} - N + \operatorname{tr}\left(\mathbf{\Sigma}_q^{-1}\mathbf{\Sigma}_p\right) + (\mu_q - \mu_p)^\top \mathbf{\Sigma}_q^{-1}(\mu_q - \mu_p)\right) \tag{2.6.3}$$

Another interesting situation where KLD finds applications is in comparing two sums of Gaussians (SOG). Unfortunately, there is no closed-form expression in this case, but an upper bound exists as proven by Runnalls in [Run07]. This result will be useful in this thesis, as shown in §14.6.2.

2.7 Bayes filtering

The Bayes rule is the grounding of an important family of probabilistic estimation techniques named *Bayes filters* which underlies many of the approaches presented in this thesis. Given a probabilistic belief about the state x of a system, Bayes filtering estimates the new belief after acquiring observations z that are directly or indirectly related to the system. Put mathematically, the rule is stated as:

$$p(x|z) = p(x)\frac{p(z|x)}{p(z)} \tag{2.7.1}$$

The belief after incorporating the observation, that is, $p(x|z)$, is referred to as the *posterior* density, while the term $p(x)$ is named the *prior* density and $p(z|x)$, which models how observations are related to the system state, is called the *observation likelihood*. The denominator $p(z)$, called the *partition function* in some contexts, represents the appropriate constant to normalize the posterior distribution and make it a

proper density function. As will be seen when discussing robot localization and SLAM (in chapters 3 and 7, respectively), this Bayes rule can be applied recursively to the sequence of variables in a dynamic system that evolves with time.

It is worthy to highlight a convention that will be used throughout this thesis: the distribution that represents the knowledge *before* incorporating the new information in the Bayes equation, that is, $p(x)$ in Eq. (2.7.1), will be always called *Bayes prior*, or just *prior* if there is no ambiguity. This remark is necessary since, strictly speaking, "priors" are distributions not conditioned to any other variable, whereas it will be quite common to find Bayes priors as $p(x|B)$, with B being previous knowledge, e.g. already incorporated in previous steps of a sequential Bayes filter.

Note that the Bayes rule states the relation between generic probability densities. Each of the existing Bayesian filters specializes in approaching the problem for a different representation of the densities, and/or for the cases of linear and non-linear models, as briefly introduced in §§2.7.1–2.7.2. A deeper discussion about Bayesian filters can be found elsewhere [Dou01, Ris04].

2.7.1 Gaussian filters

This family of Bayesian filters relies on multivariate Gaussian distributions to model the uncertainty in all the variables of both the system state and the observations. Systems with both linear transition and linear observation equations can be optimally estimated by means of the Kalman filter (KF) [Kal60], which means that the posterior converges towards the actual distribution as observations are processed.

The limitation of linear equations can be avoided by linearization, which leads to the Extended Kalman Filter (EKF) [Jul97], or by systematic sampling, which gives the Unscented Kalman Filter (UKF) [Wan00]. Other variants include the iterated EKF

(IKF) (which was proven to be equivalent to the Gauss-Newton method [Bel93]), and the Information Filter (IF) [Wal07], which maintains and updates normal distributions in their canonical form [Wu05].

2.7.2 Particle filters

The application of Monte-Carlo simulation to sets of particles with the aim of approximating Bayes filtering leads to a group of techniques generically named *particle filters* (PF) or *Sequential Monte Carlo* (SMC) estimation. The main advantages of PFs in comparison to Kalman Filters are the avoidance of linearization errors and the possibility of representing other pdfs apart from Gaussians. On the other hand, PFs require a number of samples that grows exponentially with the dimensionality of the problem.

In its simplest form, a PF propagates the samples that approximate the prior in the state space using the transition model of the system, and then updates their weights using the observation likelihood. A detailed survey of particle filtering techniques can be found in Chapter 4, where we also propose a novel particle filter algorithm.

As an additional remark, in most practical situations (where linear models are not applicable) Monte-Carlo simulation with a large enough number of samples is usually considered the standard for approximating the ground-truth distribution of the posterior.

2.8 Resampling strategies

A common problem of all particle filters is the *degeneracy of weights*, which consists of the unbounded increase of the variance of the weights $\omega^{[i]}$ with time[2]. In order to prevent this growth of variance, which entails a loss of particle diversity, one of a set of *resampling* methods must be employed.

The aim of resampling is to replace an old set of N particles by a new one with the same population size but where particles have been duplicated or removed according to their weights. More specifically, the expected duplication count of the i'th particle, denoted by N_i, must tend to $N\omega^{[i]}$. After resampling, all the weights become equal to preserve the importance sampling of the target pdf. Deciding whether to perform resampling or not is most commonly done by monitoring the *effective sample size* (ESS) [Liu96], obtained from the N normalized weights $\omega^{[i]}$ as:

$$ESS = \left(\sum_{i=1}^{N} (\omega^{[i]})^2 \right)^{-1} \quad \in [1, N] \tag{2.8.1}$$

The ESS provides a measure of the variance of the particles' weights, e.g. the ESS tends to 1 when one single particle carries the largest weight and the rest have negligible weights.

2.8.1 Overview

This section describes four different strategies for resampling a set of particles whose normalized weights are given by $\omega^{[i]}$, for $i = 1, ..., N$. The methods are explained using a visual analogy with a "wheel" whose perimeter is assigned to the different particles in such a way that the length of the perimeter associated to each particle is proportional to

[2] This variance is defined between *different executions* of the particle filter; it must remain clear that importance weights are just scalars, not pdfs.

its weight (see Figures 2.4–2.7). Therefore, picking a random direction in this "wheel" implies choosing a particle with a probability proportional to its weight. For a more formal description of the methods, please refer to the excellent paper by Douc, Cappé and Moulines [Dou05].

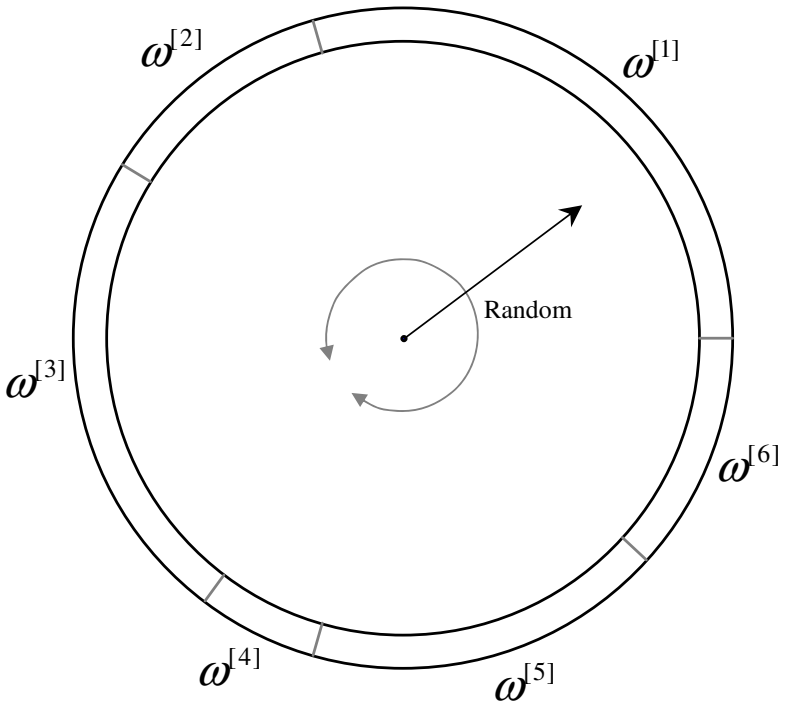


Figure 2.4: The multinomial resampling algorithm.

- **Multinomial resampling:** It is the most straightforward resampling method, where N independent random numbers are generated to pick a particle from the old set. In the "wheel" analogy, illustrated in Figure 2.4, this method consists of picking N independent random directions from the center of the wheel and taking the pointed particle. The name of this method comes from the fact that the probability mass function for the duplication counts N_i is a multinomial distribution with the weights as parameters.

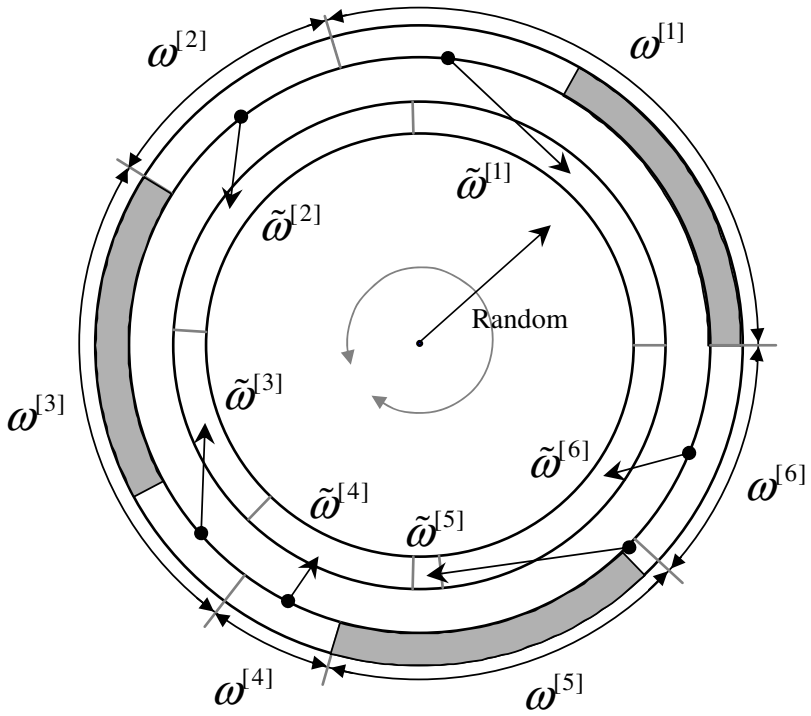


Figure 2.5: The residual resampling algorithm. The shaded areas represent the integer parts of $\omega^{[i]}/(1/N)$. The residual parts of the weights, subtracting these areas, are taken as the modified weights $\tilde{\omega}^{[i]}$.

- **Residual resampling:** This method comprises two stages, as can be seen in Figure 2.5. Firstly, particles are resampled deterministically by picking $N_i = \lfloor N\omega^{[i]} \rfloor$ copies of the i'th particle. Then, multinomial sampling is performed with the residual weights:

$$\tilde{\omega}^{[i]} = \omega^{[i]} - N_i/N.$$

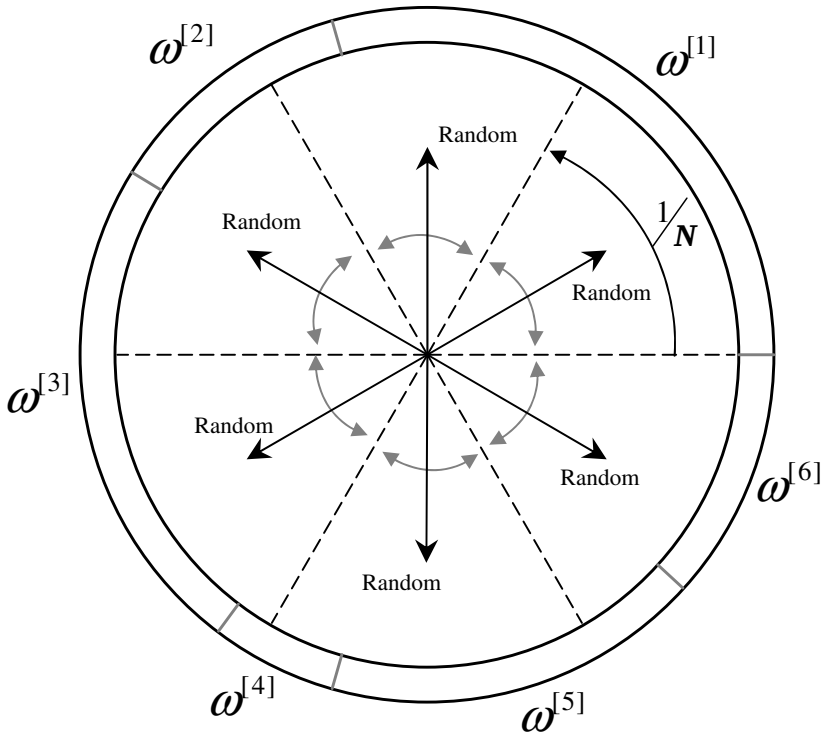


Figure 2.6: The stratified resampling algorithm. The entire circumference is divided into N equal parts, represented as the N circular sectors of $1/N$ perimeter lengths each.

- **Stratified resampling:** In this method, the "wheel" representing the old set of particles is divided into N equally-sized segments, as represented in Figure 2.6. Then, N numbers are independently generated from a uniform distribution like in multinomial sampling, but instead of mapping each draw to the entire circumference, they are mapped within its corresponding partition out of the N ones.

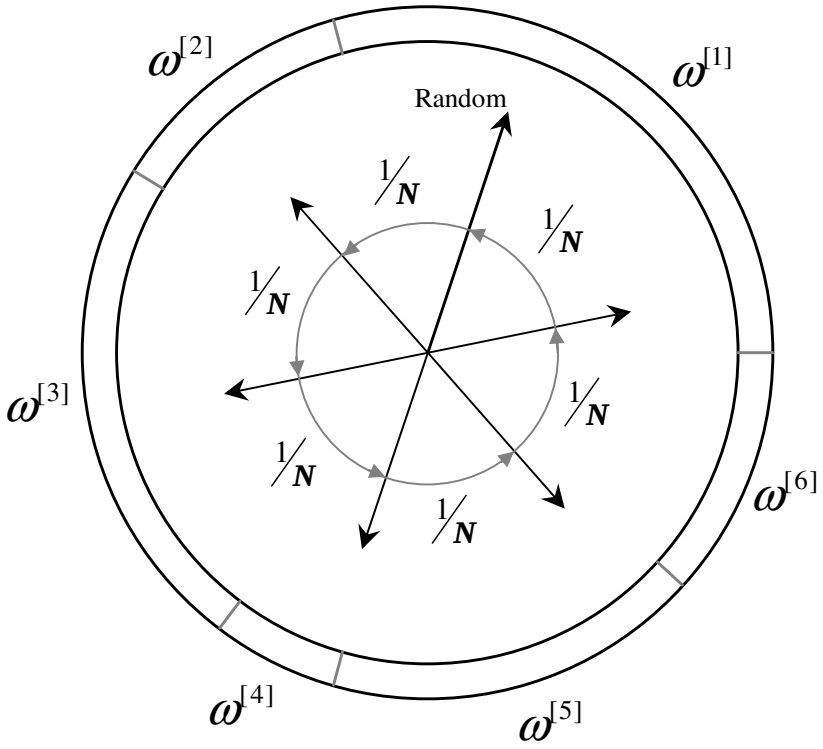


Figure 2.7: The systematic resampling algorithm.

- **Systematic resampling:** Also called *universal sampling*, this popular technique draws only one random number, i.e. one direction in the "wheel", with the others $N-1$ directions being fixed at $1/N$ increments from that randomly picked direction.

2.8.2 Comparison of the four methods

In the context of Rao-Blackwellized particle filters (RBPF) [Dou00a], where each particle carries a hypothesis of the complete history of the system state evolution, resampling becomes a crucial operation that reduces the diversity of the PF estimate for past states. In order to evaluate the impact of the resampling strategy on this

loss, the four different resampling methods discussed above have been evaluated in a benchmark [Bla09a] that measures the diversity of different states remaining after t time steps, assuming all the states were initially different.

The results, displayed in Figure 2.8, agree with the theoretical conclusions in [Dou05], stating that multinomial resampling is the worst of the three methods in terms of variance of the sample weights. Therefore, due to its simple implementation and good results, the systematic method is employed in the rest of this work when evaluating "classical" particle filters, in constrast to our new proposed algorithms where resampling is addressed differently, as explained in the corresponding chapters.

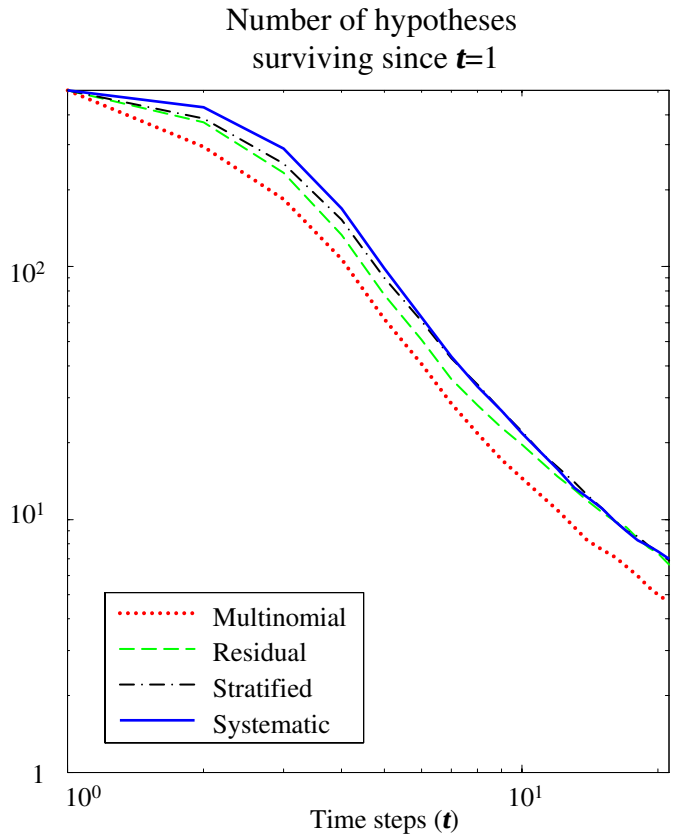


Figure 2.8: A benchmark to measure the loss of hypothesis diversity with time in an RBPF for the four different resampling techniques discussed in this section. The multinomial method clearly emerges as the worst choice.

2.9 Random sample generation

A recurrent topic throughout probabilistic mobile robotics is drawing samples from some probability distribution. Typical applications include Monte Carlo simulations for modeling a stochastic process and some steps within particle filtering.

The most basic random generation mechanism is that for uniform distributions over integer numbers, since *any* other discrete or continuous distribution can be derived from it. Due to its importance, most modern programming languages include such a uniform random generator. In the case of C or C++, the POSIX.1-2001 standard proposes a simple pseudorandom generator – the implementation of the function `rand` in the default C libraries. However, this method has its drawbacks in both efficiency and randomness [Wil92].

In this thesis it has been employed an alternative method, namely the MT19937 variation of the Mersenne twister algorithm [Mat98]. Note that although this method generates integer numbers, generating uniformly-distributed real numbers from them is trivial. An application of uniform sampling in the context of resampling has been already discussed in §2.8.

Another distribution useful in probabilistic robotics is the 1-dimensional normalized Gaussian, that is, with mean zero and unit variance, which can be obtained from uniformly-distributed numbers following the algorithm `gasdev` described in [Wil92].

Drawing samples from multivariate Gaussians is the most important sampling method in our context, since it is involved in most particle filters for robot localization and mapping (e.g. for drawing samples from the motion models). Let $\bar{\mathbf{x}}$ and $\mathbf{\Sigma}_x$ denote the mean and covariance, respectively, of the N-dimensional Gaussian from which a sample $\mathbf{x}$ will be drawn, that is,

$$\mathbf{x} \sim \mathcal{N}(\bar{\mathbf{x}}, \mathbf{\Sigma}_x) \tag{2.9.1}$$

If we obtain the N eigenvalues e_i and eigenvectors $\mathbf{v}_i$ of $\mathbf{\Sigma}_x$, the k'th element of the sample $\mathbf{x}$ is given by

$$\mathbf{x}(k) = \bar{\mathbf{x}}(k) + \sum_{i=1}^{N} \sqrt{e_i}\mathbf{v}_i(k) r_k \tag{2.9.2}$$

where each r_k is a random sample from a 1-dimensional, normalized Gaussian. This method can be seen as a change of bases from the N-hypersphere where the N independent and identically distributed (i.i.d.) random samples r_k are generated, into the orthogonal base defined by the eigenvectors of the covariance matrix $\mathbf{\Sigma}_x$, each dimension scaled by the corresponding eigenvalue.

2.10 Graphical models

Graphical models are a powerful tool where concepts from both graph and probability theories are applied in order to make efficient statistical inference or sampling in problems involving several random variables [Bis06, Jen96]. In this paradigm, a graph represents a set of random variables (the nodes) and their statistical dependencies (the edges). Graphical models have recently inspired interesting approaches in the mobile robot SLAM community [Cum08,Pas02]. Furthermore, some well-known methods such as the Kalman filter or Hidden-Markov models can be shown to be specific instances of inference on graphical models [Bis06].

Depending on the edges being directed or undirected, the resulting graph is known

as a *Bayesian network* (BN) or a *Markov random field*, respectively. In this work we are only interested in BNs, where the direction of edges can be interpreted as a *causal* relation between the variables that gives raise to a statistical dependence.

To illustrate the usage of a BN to, for example, factor a joint distribution, consider the example in Figure 2.9 with the five random variables $\{a, b, c, d, e\}$:

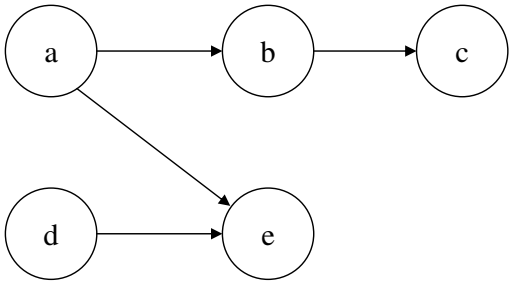


Figure 2.9: An example of a Bayesian network with five random variables.

Given the dependencies in this graph, the joint distribution of the system can be factored exploiting the *lack* of connections between some nodes. In the current example, using the Law of Total Probability we arrive at:

$$p(a, b, c, d, e) = p(a)p(d)p(b|a)p(c|b)p(e|a, d). \tag{2.10.1}$$

Another important information that can be determined from a BN is whether two variables (or sets of variables [3]) a and c are *conditionally independent* given another set of variables b, what is sometimes denoted as [Daw79]:

$$a \perp\!\!\!\perp c \mid b \tag{2.10.2}$$

This condition can be asserted from the BN by inspecting whether the fixation of the values of the variables in b leaves no connection that could carry information from

[3] An advantage of graphical models is that what is true for each node representing one random variable, also holds when the nodes represent sets or *clusters* of variables.

variables in a to those in c, or whether the arrows (in the possible paths between a and c) meet head-to-head at some intermediary node that is *not* part of b. If at least one of those two conditions holds, it is said that b *d-separates* a and c. For a more formal definition of the rules to determine d-separation, please refer to [Bis06, Rus95b, Ver90].

A pair of variables a and d (refer to Figure 2.9) can be also (unconditionally) independent, which can be denoted as $a \perp\!\!\!\perp d \mid \varnothing$ or simply $a \perp\!\!\!\perp d$. From the rules to determine d-separation, it can be deduced that two variables are independent only if all the paths between them end in a head-to-head arrow configuration (or, obviously, when there is no connection at all between them).

Summarizing for the example in Figure 2.9, it can be observed that b d-separates a and c, thus $a \perp\!\!\!\perp c \mid b$, and therefore the distribution $p(a, c|b)$ can be factored as $p(a|b)p(c|b)$. Moreover, it can be observed that a and d are (unconditionally) independent variables (since the arrows in the path $a \leftrightarrow d$ meet head-to-head), that is, $a \perp\!\!\!\perp d$, and hence $p(a, d) = p(a)p(d)$.

The application of these simplifications is at the core of many modern approaches to map building in mobile robotics, including the framework presented in Chapter 12.

Part I

Mobile Robot Localization

CHAPTER 3

OVERVIEW

This chapter briefly reviews the problem of mobile robot localization and some of the existing solutions proposed in the literature.

In general, the reported methods can be divided into those based on metric maps (e.g. sets of landmarks, occupancy grids) and those relying on topological maps (e.g. a graph of distinctive "places"). Although the latter approach has led to some successfully localization frameworks [Cum08, Kui90], the largest part of the literature focuses on pure metric methods due to their better suitability to indoor, possibly cluttered (and dynamic) spaces where a mobile robot needs an accurate pose estimation in order to perform some basic tasks such as motion planning.

Metric localization most commonly addresses the problem of *pose tracking*, where an estimate of the robot pose is sequentially updated as new data are gathered by the sensors (e.g. wheels odometry, camera images or laser scans). This situation can be characterized by the existence of a unique robot pose hypothesis relatively well

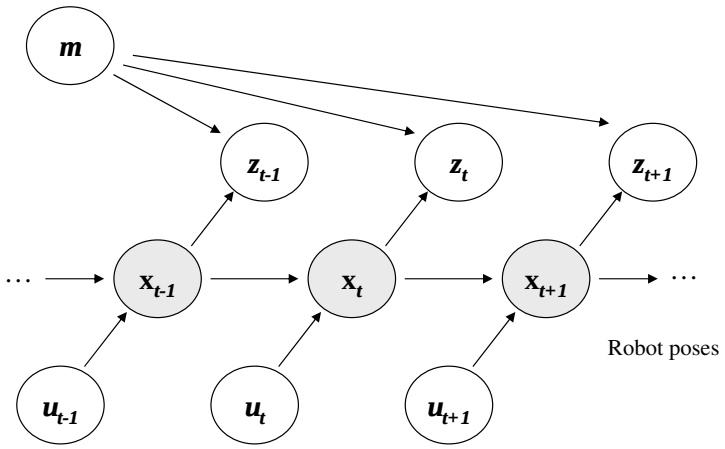


Figure 3.1: The dynamic Bayesian network for mobile robot localization, where robot poses x_t are hidden variables (represented as shaded nodes) to be estimated from actions u_t, sensor observations z_t and a model of the environment m.

localized in space. An extension of this paradigm is *global localization*, the problem of a robot "awakening" in an unknown position of the environment. In this case, a multitude of localization hypotheses must be managed simultaneously.

Bayesian filtering allows the coherent treatment of both, global localization and pose tracking in a single probabilistic framework. Although parametric filters such as multihypotheses Kalman filters have been used for this aim [Arr02a], the most common approach is particle filtering [Thr05].

Mathematically, probabilistic robot localization consists of estimating the distribution of a hidden dynamic variable x_t, standing for the robot pose at time step t, given sensor observations z_t, a map of the environment m and robot actions u_t (normally, odometry increments).

From the dynamic Bayesian network (DBN) of the problem, displayed in Figure 3.1, it is clear that the sequence of robot poses constitute a Markov process, that is, given a pose x_t, the pose at the next instant x_{t+1} is conditionally independent of all previous poses:

$$x_{t+1} \perp\!\!\!\perp x_1, x_2, ..., x_{t-1} \mid x_t. \tag{3.0.1}$$

Therefore, the problem can be addressed sequentially by estimating one robot pose at once based on the previously estimated pose, using the well-known expression:

$$p(x_t|z_{1:t}, u_{1:t}, m) \propto p(z_t|x_t, m) \int p(x_t|x_{t-1}, u_t) p(x_{t-1}|z_{1:t-1}, u_{1:t-1}, m) dx_{t-1} \tag{3.0.2}$$

The derivation of this equation is given next along a detailed description of the probabilistic rules applied in each step. Note that superscripts have been used for shortening the expressions with sequences of variables, e.g. $x^t \doteq x_{1:t}$.

$$\underbrace{p(x_t|z^t, u^t, m)}_{\text{Posterior for } t} \propto$$

Bayes on z_t

$$p(z_t|x_t, z^{t-1}, u^t, m) p(x_t|z^{t-1}, u^t, m) =$$

$z_t \perp\!\!\!\perp z^{t-1}, u^t \mid x_t, m$

$$\underbrace{p(z_t|x_t, m)}_{\substack{\text{Observation} \\ \text{likelihood}}} p(x_t|z^{t-1}, u^t, m) =$$

Law of total probability on x_{t-1}

$$p(z_t|x_t, m) \int_{-\infty}^{\infty} p(x_t|x_{t-1}, z^{t-1}, u^t, m) p(x_{t-1}|z^{t-1}, u^t, m) dx_{t-1} =$$

$x_t \perp\!\!\!\perp z^{t-1}, u^{t-1}, m \mid x_{t-1}, u_t$

$$p(z_t|x_t, m) \int_{-\infty}^{\infty} \underbrace{p(x_t|x_{t-1}, u_t)}_{\text{Motion model}} p(x_{t-1}|z^{t-1}, u^t, m) dx_{t-1} =$$

$x_{t-1} \perp\!\!\!\perp u_t \mid \varnothing$

$$p(z_t|x_t, m) \int_{-\infty}^{\infty} p(x_t|x_{t-1}, u_t) \underbrace{p(x_{t-1}|z^{t-1}, u^{t-1}, m)}_{\text{Posterior for } t-1} dx_{t-1}$$

In the next chapter it is explored a novel algorithm for effective Bayesian filtering and its applications to mobile robot localization. Then, Chapter 5 discusses a proposal

for a new observation model useful for localization with accurate laser range scanners in dynamic environments. Finally, we also address in Chapter 6 the issue of improving a robot probabilistic motion model by means of fusing different proprioceptive sensors.

CHAPTER 4

OPTIMAL PARTICLE FILTERING FOR NON-PARAMETRIC OBSERVATION MODELS

4.1 Introduction

Sequential estimation of dynamic, partially observable systems is a problem with numerous applications in a wide range of engineering and scientific disciplines. The state-space form of this problem consists of iteratively tracking the state of a system at discrete time steps given the system transition and observation models and a sequence of observations. In a probabilistic framework, sequential Bayesian filtering represents

an effective solution [Liu98, Dou01, Ris04].

In the scope of mobile robotics there are two prominent applications of Bayesian sequential estimation that have received a huge attention by the research community in the last decade, namely localization and simultaneous localization and map building (SLAM) [Thr01b, Thr05, Est05, Gut99, Fox99b, Dis01, Hah03, Thr02, Gri07b]. As discussed in Chapter 3, the former consists of estimating the pose of a mobile robot within a previously known environment, whereas in SLAM a map of the environment is estimated from scratch while performing self-localization.

In both cases the choice for the representation of the environment determines which Bayesian estimation method can be applied. For example, landmark maps can be modeled by multivariate Gaussian distributions with Gaussian observation models that can be obtained by solving the data association problem [Dis01, Dav07]. Therefore, SLAM with landmark maps can be solved through Gaussian filters such as the EKF [Jul97] or the UKF [Wan00]. However, for other types of map representations, like occupancy grid-maps [Mor85, Thr03], these filters are not applicable, forcing a sample-based representation of probability densities and sequential estimation carried-out via Monte-Carlo simulations (the filtering algorithms becomes a particle filter [Dou01]).

In this chapter we focus on occupancy grids as map model, although the described method, published in [Bla08h], can be also applied to other maps compatible with a sample-based representation of probability distributions (e.g. gas concentration maps [Lou07], topological maps [Ran06]). Among the advantages of mapping with occupancy grids we find the precise dense information they provide and the direct relation of the map with the sensory data, which avoids the problem of data association that is present in landmark maps [Nei01]. Their main drawback is that the observation likelihood model for grid maps can be evaluated only pointwise in a non-parametric form [Thr05, Thr01a], in contrast to analytical models available for landmark maps

[Dav07, Dis01] that enable the direct computation of the optimal probability densities [Dou00b].

Provided that we are able to draw samples according to the system transition model (the robot motion model in our case) and to pointwise evaluate the observation model, we can sequentially solve localization and SLAM through one of the most basic particle filter algorithms: the Sequential Importance Sampling (SIS) filter [Rub87], subsequently modified to account for the particle depletion problem [Aru02] by means of a resampling step, leading to the SIS with resampling (SIR) filter [Rub88, Gor93]. However, the behavior of these algorithms is greatly compromised by peaky sensor models and outliers, which make most of the particles to be discarded in the resampling step, which leads to *particle impoverishment* or even to the divergence of the filter. In mobile robotics, this issue typically arises in vehicles equipped with low-noise sensors such as laser range finders [1].

A theoretical approach that enables the efficient representation of probability densities through perfectly distributed particles (and thus, avoiding particle depletion as much as possible) was proposed by Doucet *et al.* [Dou00b], consisting of an optimal proposal distribution from which to draw samples at each time step. However, a direct application of this approach requires an observation model with a parametric distribution (from which random samples could be drawn), whereas, as mentioned above, for grid maps we can evaluate it only pointwise [Thr05].

This chapter describes a new particle filter algorithm that, given the same requirements as the original SIS and SIR algorithms, dynamically generates the minimum number of particles that *best* represent the true distribution within a given bounded error, thus providing optimal sampling. The method is grounded on previous works

[1] Chapter 5 presents another solution to the problem of peaky likelihood models by means of a new likelihood model, whereas in this chapter the focus is on doing Bayesian filtering for any sensor model, no matter how peaky it could be.

related to optimal sampling [Dou00b, Dou00a], auxiliary particle filters (APF) [Pit99], rejection sampling [Liu98], and adaptive sample size for robot localization [Fox03]. All these ideas will be discussed throughout subsequent sections.

When applied to mobile robots, the proposed method represents important improvements with respect to previous algorithms for efficient localization and grid map building:

- No Gaussian approximations are assumed for the generation of new particles, which is the case of previous PF works (e.g. [Mon03a, Gri07b]).

- Our method is based on the formulation of a general particle filter, and does not depend on the reliability of scan matching to approximate the observation likelihood in the case of range scans, as previous works do. Approximating the peak of the posterior distribution by a Gaussian centered at the result of scan matching actually hides the true robot pose distribution, and may lead to filter divergence if the scan matching is poor or fails, as pointed out in [Mon03a, Gri07b].

It should be stressed that, like other particle filter algorithms, our proposal should be used only when either the system models are non-linear or the filtered distributions or the observation model cannot be approximated well by Gaussians. Otherwise, the very well-known Kalman-like filters [Jul97, Wan00] are more efficient and convenient.

In the next section, we review previous particle filter algorithms that have been applied to robotics. Our proposal is introduced in section 4.3, and its computational complexity analysis is presented in section 4.4, with experimental results presented next.

4.2 Background

In this section we review the underlying ideas of Monte Carlo methods for sequential Bayesian filtering, focusing on the applications of particle filters to robot localization and SLAM. For a good introduction to particle filters in tracking problems the reader can refer to [Aru02], while a more exhaustive review of theoretical advances in the field can be found in [Dou01, Ris04].

With subtle differences, the solutions to both localization and SLAM include the estimation of the posterior distribution of the robot poses up to the current instant of time, given the whole history of available data. Let $x^t = \{x_1, ..., x_t\}$ denote[2] the sequence of robot poses (the robot *path*) up to time step t. Then, the posterior of the robot pose can be computed sequentially by applying the Bayes rule:

$$p(x^t|z^t, u^t) \propto \overbrace{p(z_t|x^t, u^t)}^{\text{Observation likelihood}} \overbrace{p(x^t|z^{t-1}, u^t)}^{\text{Prior}} \tag{4.2.1}$$

where the z^t and the u^t represent the sequences of robot observations and actions, respectively. In the case of localization, we will be interested just in the last robot pose instead of the whole path, which it is the case in SLAM.

Under the assumptions of Gaussian distributions and linear systems, the Kalman filter [Kal60] offers a closed-form, optimal solution to Eq. (4.2.1). Several improvements have been proposed to overcome the constraint of linear system models, leading to the Extended Kalman Filter (EKF) [Jul97] (and its dual, the Extended Information Filter (EIF) [Thr04]), the Unscented Kalman Filter (UKF) [Jul02, Wan00], and other higher-order approximations [Ten03]. The EKF has been the predominant approach to localization and SLAM for almost a decade [Dis01]. However, drawbacks like the lack

[2]Remember that, for the sake of readability, sequences of variables over time are denoted by the last time step in the sequence placed as a superscript.

of multihypothesis support in these Gaussian filters led to the popularization of particle filters for global localization [Fox99a] and for mapping [Gri07a, Mon02a, Mur99].

As opposed to parametric probability distributions (e.g. Gaussians and sum of Gaussians), in a particle filter the estimated distribution of the pose (and the map in the case of SLAM) is represented by a finite set of hypotheses, or *particles*, which are weighted according to *importance sampling*. As mentioned, the simplest particle filter algorithm is the SIS filter [Rub87], which is discussed next in the context of robot localization. Concretely, let $\{x^{t,[i]}\}_{i=1}^{M_t}$ denote a set of M_t particles, where each of the samples $x^{t,[i]}$ represents a hypothesis for the robot path up to time step t, denoted as $x^t = \{x_1, ..., x_t\}$. These particles are approximately distributed according to the posterior pdf, that is:

$$x^{t,[i]} \sim p(x^t|z^t, u^t) \qquad , i = 1, ..., M_t \tag{4.2.2}$$

Virtually all previously existing particle filter techniques rely on M_t being constant for all time steps t (we can find an exception in the work by Fox in [Fox03]). Since the particles in Eq. (4.2.2) will be not, in general, distributed exactly according to the true posterior, they are weighted by the so called *importance weights* $\omega_t^{[i]}$ in order to obtain at least an unbiased estimation of the density. The SIS algorithm consists of simulating the Bayes update in Eq. (4.2.1) by drawing samples for the new robot pose x_t from some *proposal distribution* [Dou00a]:

$$x_t^{[i]} \sim q(x_t|x^{t-1,[i]}, z^t, u^t) \tag{4.2.3}$$

and then updating their weights by:

$$\omega_t^{[i]} \propto \omega_{t-1}^{[i]} \frac{p(z_t|x_t, x^{t-1,[i]}, z^{t-1}, u^t)p(x_t|x_{t-1}^{[i]}, u_t)}{q(x_t|x^{t-1,[i]}, z^t, u^t)} \tag{4.2.4}$$

The simplest choice for the proposal distribution $q(\cdot)$ is obviously the robot motion model applied to the previous state estimate for $t-1$ – that is, the *prior* term in Eq. (4.2.1). In this thesis, we will refer to this choice as the *standard proposal*. *Only* in this case, widely employed in robotics [Fox99a, Fox03, Mon02a], the weight update in Eq. (4.2.4) simplifies to the previous weights times the evaluation of the observation model at each particle, that is:

$$\omega_t^{[i]} \propto \omega_{t-1}^{[i]} p(z_t|x^{t,[i]}, z^{t-1}, u^t) \tag{4.2.5}$$

Note how the SIS filter requires only the ability of drawing samples from the robot motion model and evaluating the observation likelihood pointwise. However, in spite of its simplicity, the SIS filter presents some important drawbacks that limits its practical utility: it has been demonstrated that the variance of the weights increases over time [Dou00a], which eventually leads to the degeneracy of the filter. This is the reason for the introduction of the SIS with resampling (SIR) algorithm [Gor93], where a resampling step is introduced to replace those particles with low weights by copies of more likely particles.

Regarding the specific case of robot SLAM, Rao-Blackwellized Particle Filters (RBPF) are a practical solution for simultaneously estimating both the robot path and the map [Mur99]. These particle filters have been used for both landmark maps (FastSLAM [Mon02a]) and occupancy grids [Gri07a].

The above particle filter algorithms are strongly influenced by the choice of the proposal distribution $q(\cdot)$, which must not have any special relationship to the transition

model, in spite of the widespread usage of the standard proposal (mainly due to the simplified formula then obtained for updating the weights). The larger the mismatch between the proposal $q(x_t|...)$ and the observation likelihood $p(z_t|x_t, ...)$, the more particles are wasted in non-relevant areas of the state space x_t. In particular, this is the case for mobile robots equipped with accurate sensors like laser range finders [Gri07a].

An improved approach was presented by Pitt and Shephard in [Pit99] through the Auxiliary Particle Filter (APF), which has also been applied to robot localization [Vla02]. In an APF, the process of drawing particles is split into two steps. Firstly, each particle in the previous time step is assigned a measure of its predicted accordance with the most recent observation, and then only those particles that obtain high weights are propagated. Thus, a one-step look ahead resampling is introduced at each step t in this filter. In general, an APF outperforms traditional filters in the cases of peaky observation models or outliers, reducing the number of wasted particles. However, the particles are also propagated using the standard proposal distribution, which is a sub-optimal solution, in contrast to the approach discussed next.

It has been demonstrated by Doucet *et al.* [Dou00b] that the variance of the particle weights is minimized by choosing a so-called *optimal proposal distribution* , which incorporates the information of the most recent observation while propagating particles. For landmark maps there exists a closed-form solution to this equation, which has been reported as FastSLAM 2.0 [Mon03a]. However, for other map representations like occupancy grids, parametric observation models are not available and the approach cannot be directly applied.

A solution recently proposed by Grisetti *et al.* [Gri07a, Gri07b] for grid maps overcomes this limitation by approximating the sensor model with a Gaussian whose mean is obtained by scan matching over the grid map, and thus leading to a parametric formulation where the optimal distribution can be applied to. This approximation has

Table 4.1: A comparison of existing Bayesian filtering algorithms

Proposal distribution	**System models**	**Algorithms**
–	Linear Gaussian	Kalman Filter [Kal60]
–	Non-Linear Gaussian	EKF [Jul97], UKF [Wan00]
Standard	Non-Linear Non-Gaussian	SIR [Gor93], APF [Pit99], RBPF [Mur99], FastSLAM [Mon02a]
Optimal	Non-Linear Gaussian	FastSLAM 2.0 [Mon03a], Grisetti *et al.* [Gri07b, Gri07a]
Optimal	Non-Linear Non-Gaussian	Optimal PF (Proposed in this thesis)

demonstrated its practical utility allowing the efficient mapping of large environments. However, we should highlight some drawbacks of this approach. Firstly, the observation likelihood may not be appropriately approximated by a Gaussian in many situations, in which case the posterior distribution would be severely distorted. Even in those cases where the observation likelihood resembles a Gaussian, it cannot be proven that the mean value of the posterior could match well that obtained from scan matching. Actually, there are some practical situations where scan matching techniques fail. It has been proposed to discard the information of the problematic observations [Gri07a], but observe that we could obtain a more precise posterior by integrating all the available information. Secondly, the prior distribution for each time t (given by $p(x^t|z^{t-1}, u^t)$ when using the standard proposal) is ignored due to its inaccuracy in comparison to the observation likelihood. In contrast, in the exact computation of the posterior, this prior distribution (computed from the motion model) would provide valuable information when facing ambiguous sensor measurements, e.g. a robot in a populated environment where people block the scanner.

We present in Table 4.1 a classification of the methods discussed in this section, where it also appears the proposed method for comparison. It must be kept in mind that, although the present discussion is centered on localization and SLAM, the new filtering algorithm presented in this chapter can be applied to any other estimation problem where non-parametric observation models appear. Just as some examples, in the robotics and computer vision literature we can find several applications of particle filters for tracking people [Cho01a, Mon02b, Sch01] or arbitrary objects on a sequence of images [Num03, Oku04].

4.3 Particle filtering with the optimal proposal

4.3.1 Definitions

It has been shown that the proposal distribution $q(\cdot)$ that minimizes the variance of the particle weights for any generic particle filter, called optimal proposal distribution, is given by [Dou00b]:

$$x_t^{[i]} \sim q(x_t|x^{t-1,[i]}, z^t, u^t) = p(x_t|x^{t-1,[i]}, z^t, u^t)$$

which can be expanded using the Bayes rule as:

$$q(x_t|x^{t-1,[i]}, z^t, u^t) = \frac{p(z_t|x_t, x^{t-1,[i]}, z^{t-1}, u^t)p(x_t|x^{t-1,[i]}, z^{t-1}, u^t)}{p(z_t|x^{t-1,[i]}, z^{t-1}, u^t)} \tag{4.3.1}$$

For mobile robots, this proposal requires drawing samples from the product of the transition (robot motion) and observation models, both terms found in the numerator of Eq. (4.3.1). Since the system state for the last time step (x_t) does not appear in

the denominator, it becomes a constant value μ for each particle i. Therefore, to draw samples from that optimal proposal is equivalent to drawing them from:

$$x_t^{[i]} \sim \frac{1}{\mu} \overbrace{p(z_t|x_t, x^{t-1,[i]}, z^{t-1}, u^t)}^{\text{Observation model}} \overbrace{p(x_t|x^{t-1,[i]}, z^{t-1}, u^t)}^{\text{Transition model}} \tag{4.3.2}$$

By replacing this optimal proposal into the general equation for the weight update in a SIS filter, in Eq. (4.2.4), we obtain the recursive form:

$$\omega_t^{[i]} \propto \omega_{t-1}^{[i]} p(z_t|x^{t-1,[i]}, z^{t-1}, u^t) \tag{4.3.3}$$

The presented approach will allow us to generate samples exactly distributed according to the density in Eq. (4.3.1), that is, to draw samples from the optimal proposal $q(\cdot)$. Simultaneously, our method dynamically adapts the number of samples to assure the best possible representation of the true posterior at each moment.

In order to avoid the problem of particle depletion, we can find two different approaches in the literature. The first one is to resample particles at every time step of the filter. Another solution consists of resampling only when a measure of the representativeness of the samples is below a given threshold [Rub88]. The first approach is employed in this chapter to derive our optimal filtering algorithm. This generic optimal filter fits perfectly to the problem of mobile robot localization. It is left for Chapter 8 to address the required modifications to make the algorithm more suitable to the higher dimensionality of the SLAM problem.

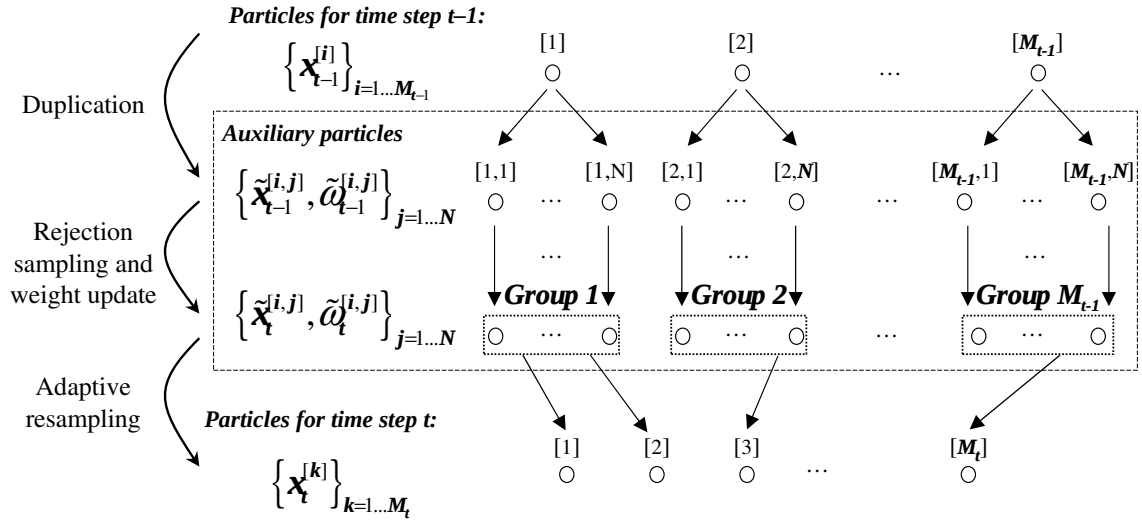


Figure 4.1: The theoretic model of our optimal particle filter. A given set of M_{t-1} particles is first replicated into a larger set of auxiliary particles, which are then propagated according to the optimal proposal distribution (obtained by rejection sampling). Then, a resampling stage with an adaptive sample size chooses the final set of M_t samples from the updated auxiliary particles, taking each one of them with a probability proportional to its weight. As a result, all the final particles have equal importance weights (omitted in the graph for this reason).

4.3.2 Derivation of the optimal filter algorithm

In the following we derive the algorithm for generating a dynamically-sized set of samples according to the exact posterior being estimated, and thus leaving the non-parametric and finite nature of the filter as the only source of errors in the estimation of the true posterior. To clarify the exposition we have summarized the process graphically in Figure 4.1.

We start by assuming that at filter time $t-1$, a set of M_{t-1} particles $x_{t-1}^{[i]}$ is available which are *exactly* distributed according to the posterior, that is:

$$x_{t-1}^{[i]} \sim p(x_{t-1}|z^{t-1}, u^{t-1}) \tag{4.3.4}$$

Since these samples are optimally distributed, all of them will have equal importance

weights, and so those weights can be omitted. The assumption of perfectly distributed particles for the previous time step could be a problem for the first iteration of the filter, but notice that at this first step the best we can do is to simply consider some prior of the state. Typical assumptions in the literature for this initial belief prior include uniform or Gaussian distributions, depending on the available information and the specific problem.

Now we introduce a set of *auxiliary particles* $\tilde{x}_{t-1}^{[i,j]}$ with associated importance weights $\tilde{\omega}_{t-1}^{[i,j]}$, such that:

$$\begin{aligned} \tilde{x}_{t-1}^{[i,j]} &= x_{t-1}^{[i]} \qquad , j = 1, ..., N \\ \tilde{\omega}_{t-1}^{[i,j]} &= \frac{1}{NM_{t-1}} \end{aligned} \tag{4.3.5}$$

That is, we simply replicate N times each particle $x_{t-1}^{[i]}$, assigning equal weights to all of them. Notice that this process does not modify the sample-based estimation of the posterior, since each particle i has the same number of replications. We will use this set of auxiliary particles just as an auxiliary computation element: in practice only a few of them will be actually generated, as it will become clear below. Therefore, the value N is left undefined here, although it is convenient to think of it as a large value, ideally infinity.

Now, the auxiliary particles $\tilde{x}_{t-1}^{[i,j]}$ are propagated into $\tilde{x}_t^{[i,j]}$ following the optimal proposal in Eq. (4.3.2) in order to obtain a large amount (ideally infinity) of optimally distributed particles from which we will finally keep only the required ones for providing a good representation of the posterior. This reduction is achieved by resampling the set of auxiliary samples $\tilde{x}_t^{[i,j]}$ to generate the new set $x_t^{[k]}$.

The key point that allows us to directly generate the optimally distributed particles without computing all the auxiliary ones is that *all* the auxiliary particles $\tilde{x}_t^{[i,j]}$ that

can be traced back to a given sample $x_{t-1}^{[i]}$ *have identical weights*. This property follows from the fact that the concrete value of the particle at time step t does not appear in the computation of the new weights, as can be seen in Eq. (4.3.3) – this is the fundamental idea that motivated the development of the whole algorithm. These groups of equally-weighted samples are schematically represented in Figure 4.1.

In order to generate the particles $x_t^{[k]}$, the (potentially infinite) set of auxiliary particles for time step t must be resampled. Similarly to auxiliary particle filters [Pit99], we perform this by drawing indexes i of particles for the previous time step. Each index i has a probability of being selected proportional (due to a constant for normalization) to the weights $\tilde{\omega}_t^{[i,j]}$, computed as (see Eq. (4.3.3)):

$$\tilde{\omega}_t^{[i,j]} \quad = \quad \tilde{\omega}_{t-1}^{[i,j]} p(z_t | x^{t-1,[i]}, z^{t-1}, u^t) \tag{4.3.6}$$

The superindices j can be dropped since, as mentioned above, the weights only depend on the value of the original particle $x_{t-1}^{[i]}$, thus:

$$\tilde{\omega}_t^{[i]} \quad = \quad \tilde{\omega}_t^{[i,\cdot]} = \tilde{\omega}_{t-1}^{[i]} p(z_t | x^{t-1,[i]}, z^{t-1}, u^t) \overbrace{\propto}^{\text{Eq. 4.3.5}} p(z_t | x^{t-1,[i]}, z^{t-1}, u^t) \tag{4.3.7}$$

The right hand term (*a priori* likelihood of the observation z_t given all knowledge up to $t-1$) can be expanded using the law of total probability:

$$p(z_t | x^{t-1,[i]}, z^{t-1}, u^t) = \int_{-\infty}^{\infty} p(x_t | x_{t-1}^{[i]}, u_t) p(z_t | x_t, x^{t-1,[i]}, z^{t-1}) dx_t \tag{4.3.8}$$

The terms that appear inside the integral above are the system transition and observation models, respectively, and therefore the integral has no closed-form solution in a system with non-parametrical models. But since we are assuming in this work

that we can draw samples from the system transition model and evaluate pointwise the observation model, a Monte-Carlo approximation $\hat{p}(z_t|\cdot) \approx p(z_t|\cdot)$ can be obtained:

$$\hat{p}(z_t|x^{t-1,[i]}, z^{t-1}, u^t) = \frac{1}{B}\sum_{n=1}^{B} p(z_t|x_t^{[n]}, x^{t-1,[i]}, z^{t-1}) \tag{4.3.9}$$

with B samples $x_t^{[n]}$ generated according to the system transition model, e.g. the robot motion model for localization and SLAM. The number B is a parameter of our algorithm, and is typically well set in the range 10 to 200 depending on the specific problem addressed by the filter (obviously, the larger the better the Monte-Carlo approximation above, at the cost of a higher computational time).

Once the weights $\tilde{\omega}_t^{[i]}$ have been approximated using Eq. (4.3.9), we proceed with the resampling of the auxiliary particles. We draw a set of indexes i, and for each one, a new optimal particle $x_t^{[k]}$ is generated by taking the value of *any* auxiliary particle in the i'th group, since all of them have equal probability of being selected in the resampling. Notice that this operation avoids the potentially infinite number of auxiliary samples, without sacrificing their optimality.

Therefore, the new optimal particle $x_t^{[k]}$ is a copy of $\tilde{x}_t^{[i,j]}$ (whose computation is discussed next), where the value of j does not need to be specified. Also, notice that the importance weights of these final particles $x_t^{[k]}$ can be ignored, since particles obtained by resampling all have exactly the same weights.

It remains to be explained how to compute the concrete value of the auxiliary particles $\tilde{x}_t^{[i,j]}$ for some certain value of i. We employ here the rejection sampling technique to draw from the product of the transition and observation densities – refer to Eq. (4.3.2). Basically, this technique consists of generating samples $x_t^{[k]}$ following one of the terms of the product (the transition model in our case), and accepting the sample with a probability Δ proportional to the other term (the observation model) [Liu98]:

$$\Delta = \frac{p(z_t|x_t^{[k]}, x^{t-1,[i]}, z^{t-1}, u^t)}{\hat{p}_{max}(z_t|x_t, x^{t-1,[i]}, z^{t-1}, u^t)} \tag{4.3.10}$$

We must remark that this technique has a stochastic complexity (a random execution time), as discussed in more detail in section 4.4. The only quantity required to evaluate Eq. (4.3.10) is the maximum value of the observation model $\hat{p}_{max}(z_t|\cdot)$. Note that this value can be estimated simultaneously to the Monte-Carlo approximation in Eq. (4.3.9) for the same set of samples $x_t^{[n]}$, thus it does not imply further computational cost.

Up to this point we have shown how to generate one particle $x_t^{[k]}$ according to the true posterior, given the set of particles for the previous time step. The method can be repeated an arbitrary number of times to generate the required number of particles M_t for the new time step t. To determine this dynamic sample size we propose to integrate here the approach introduced by Fox in [Fox03], which is based on the Kullback-Leibler divergence (KLD) (see §2.6). As that work shows, the minimum number of particles M_t to assure that the KLD between the estimated and the real distributions is kept below a certain threshold ϵ with a probability $1 - \delta$, is given by:

$$M_t = \frac{1}{2\epsilon}\chi^2_{l-1,1-\delta} \tag{4.3.11}$$

where $\chi^2_{k,c}$ stands for the c'th quantile of the chi-square distribution with k degrees of freedom. In this approach, called *KLD-sampling*, the state space of the robot is divided into a regular grid, and l represents the number of bins from that grid occupied by at least one particle. Please, refer to [Fox03] for further details.

To summarize this subsection, we present an algorithmic description of the overall method in the Algorithm 1.

Algorithm 1 optimal_particle_filter $\{x_{t-1}^{[i]}\}_{i=1}^{M_{t-1}} \rightarrow \{x_t^{[k]}\}_{k=1}^{M_t}$

for all particles $x_{t-1}^{[i]}$ **do**
  for $n = 1$ to B **do** // *Generate a set of B samples from the transition model*
    $x_t^{[n]} \sim p(x_t|x_{t-1}^{[i]}, u_t)$
  end for
  Use the $B \cdot M_{t-1}$ samples to compute $\hat{p}(z_t|\cdot)$ and $\hat{p}_{max}(z_t|\cdot)$
  Compute $\tilde{\omega}_t^{[i]}$ using Eq. (4.3.7)–(4.3.9)
end for
$k \leftarrow 1$
repeat
  Draw an index i with probability proportional to $\tilde{\omega}_t^{[i]}$.
  repeat // *Generate a new sample by rejection sampling*
    $x_t^{[k]} \sim p(x_t|x_{t-1}^{[i]}, u_t)$ // *Draw a candidate sample from the transition model*
    Compute Δ through Eq. (4.3.10)
    $a \sim \mathcal{U}(0, 1)$ // *Draw a random uniform sample*
  until $a < \Delta$// *Candidate is accepted with a probability of Δ*
  $k \leftarrow k + 1$
until $k = M_t$// *Number of samples determined by KLD-sampling, see Eq. (4.3.11) [Fox03]*

A final issue must be remarked about the implementation of this algorithm (or any other generic particle filter algorithm): due to the large dynamic ranges that can be found in particle weights and likelihood values, it is common practice to employ a *logarithmic scale*. Apart from the advantage of enjoying a huge increase in the dynamic range of particle weights, an additional advantage is that multiplying a weight with an observation likelihood becomes a sum, a more efficient operation than multiplication in all computer architectures. The only drawback of the logarithmic representation is that extracting the average of a set of logarithmic likelihood values, required while computing $\hat{p}(\cdot)$ in Eq. (4.3.9), forces us, in a naive implementation, to recover the linear weights, leading to a severe risk of numeric overflow. As a workaround, a numerically-stable method for that end is proposed in Appendix B.1.

4.3.3 Comparison to Other Methods

We discuss next an example that illustrates the differences between previous methods and our algorithm for optimal filtering. We have considered for the example a one-dimensional linear system with Gaussian transition and observation models. The purpose of using such a simple system is that of contrasting the output of the different particle filters with the analytical solution from a Kalman filter which provides us the exact posterior. The exact parameters used in this simulation are:

1. Initial belief for the state x: Gaussian centered at 0.5, with sigma of 0.4.

2. Transition model: Displacement of $\Delta x = 1$ with an additive Gaussian noise with sigma 0.2.

3. Observation likelihood: The observation z, which directly measures the state x, was in the case $z = 2$. Its additive Gaussian noise has a sigma of 0.1.

Notice that the prior before applying Bayes, i.e. taking into account the initial belief and the transition model, can be modeled as a Gaussian with sigma equal to $\sqrt{0.4^2 + 0.2^2} \approx 0.45$, which is a large uncertainty in contrast to that of the observation. Therefore, the situation being simulated is that of an observation model much more peaked than the prior before applying the Bayes rule (in mobile robotics this may represent a poor motion model, such as odometry, and a very precise sensor, such as a laser scanner).

The top graphs of Figure 4.2 represent the location and weights of the obtained particles with three different algorithms: a SIR filter with a standard proposal distribution [Rub88], the auxiliary particle filter (APF) proposed by Pitt and Shephard in [Pit99], and our method.

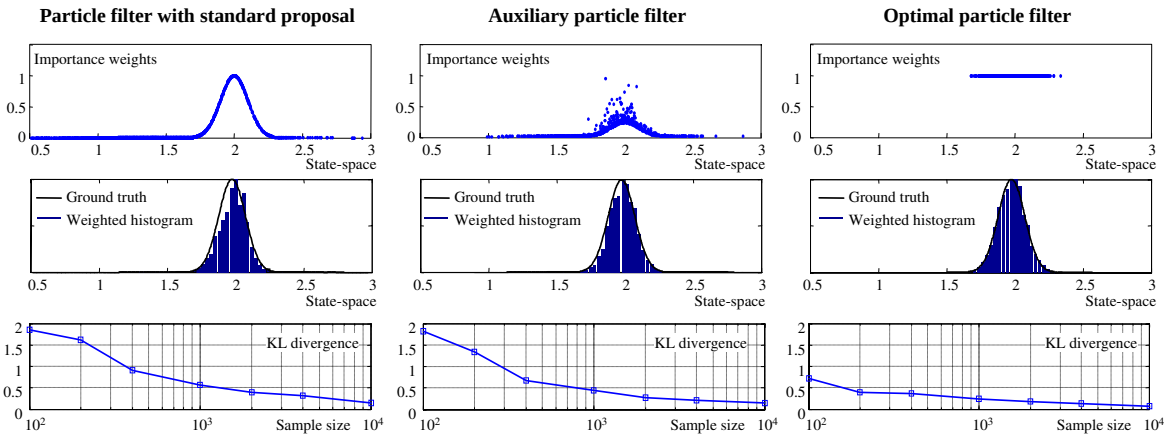


Figure 4.2: A comparison of our method to other two particle filters for a linear, Gaussian system. The top row shows the obtained particles and weights for each algorithm. Below the weighted histograms of the samples is compared to the exact Gaussian density being estimated, and it can be seen how our optimal particle filter is the one that best approximates it. To measure this similarity to the real density, the third row shows the Kullback-Leibler distance between the real and the estimated distributions for different sample sizes. Observe how our method achieves a lower distance (a higher similarity) even for a few particles.

We can observe how the standard proposal leads to many of the particles being wasted in non relevant areas of the state space, where they are assigned negligible importance weights. The APF introduces a great improvement in this sense, and particles are more concentrated in the area of interest. However, in that case the weights still contain a clearly perceptible variance. In contrast to both, our optimal algorithm generates particles distributed exactly according to the true posterior, thus they all have the same weights.

To measure the accuracy of each particle filter, we have reconstructed the estimated densities from the particle-based representation by means of weighted histograms, which are shown in the middle row of Figure 4.2 along with the ground-truth solution from the Kalman filter. To evaluate each one, we have computed the Kullback-Leibler distance (KLD) between the ground-truth and the estimated posteriors for a range of sample sizes (we have disabled here the capability of automatically determining the

sample size in our method for comparison purposes). The average KLD for 1000 realizations, shown in the bottom row of Figure 4.2, confirms that our approach gives estimations closer to the actual posterior (with less particles) than previous methods.

4.4 Complexity Analysis

In this section we analyze rigorously the computational time complexity of our optimal filter, using as a guideline the line numbers of the pseudo-code description in Algorithm 1. Recall that we defined M_{t-1} and M_t as the number of particles in the current and the previous time steps, while B represents the fixed number of auxiliary samples employed in the Monte Carlo approximation in Eq. (4.3.9). We detail next the contributions of the individual operations in our algorithm to the overall execution time of one complete filter step:

- Estimation of $\hat{p}(z_t|\cdot)$ and $\hat{p}_{max}(z_t|\cdot)$ (Lines 1–7): Here the observation model is evaluated B times for each particle in $t-1$, thus the complexity becomes $\mathcal{O}(\alpha B M_{t-1})$, where α denotes the constant time factor associated to a single pointwise evaluation of the observation model.

- Determination of M_t by means of KLD-sampling (Line 17): In principle, the most time consuming part of this method is counting the bins in the state space that hold at least one particle. However, this can be reduced to a constant time operation (with duration β) by implementing the bin counters as a simple array, as suggested in [Fox03]. Thus, the complexity of the step is $\mathcal{O}(\beta M_t)$. More memory efficient methods could be employed (such as keeping an ordered list of occupied bins) at the cost of a higher time complexity.

- Draw index samples i (Line 10): This operation requires the computation of the cumulative density function (CDF) of the particle weights, then look up (with $\mathcal{O}(\gamma M_{t-1})$) a given random value to find the corresponding index to each of the M_t particles. Thus its overall complexity becomes $\mathcal{O}(\gamma M_t M_{t-1})$.

- Rejection sampling (Lines 11–15): If we denote as R the number of trials required to get an accepted sample in each of the M_t iterations, we have a complexity of $\mathcal{O}(\alpha R M_t)$ where α is included since the observation model is evaluated at every trial. Since R is actually a random variable this operation has a non-deterministic time complexity (discussed below).

For the applications stressed in this work, i.e. mobile robots with laser scanners and occupancy grid maps, the overall complexity is strongly determined by the number of times the observation model is evaluated, since α will be usually larger than the other time constants. From the individual complexities described above we conclude that this number is of the order $\mathcal{O}(BM_{t-1} + RM_t)$. The amount of particles, M_{t-1} and M_t, will remain approximately constant for localization, whereas in SLAM the sample size increases as the robot explores long path without closing a loop, decreasing after closing them (see §8.4). Thus, any bound to the computation time limits the size of the loops our algorithm can process in real-time, just as also happens in EKF-based methods. A promising approach to overcome these limitations is to consider hierarchical (or hybrid) map representations [Bla07b, Bos04, Est05], a topic explored in part III of this thesis. The other values that determine the performance of our algorithm are B and R. Since B is a fixed parameter, we will focus next on the factors that determine the random variable R, the number of trials required to accept one sample in rejection sampling.

Firstly, we can model rejection sampling as a Bernoulli process, since each trial consists of an independent test with just two possible outcomes: acceptance or rejection.

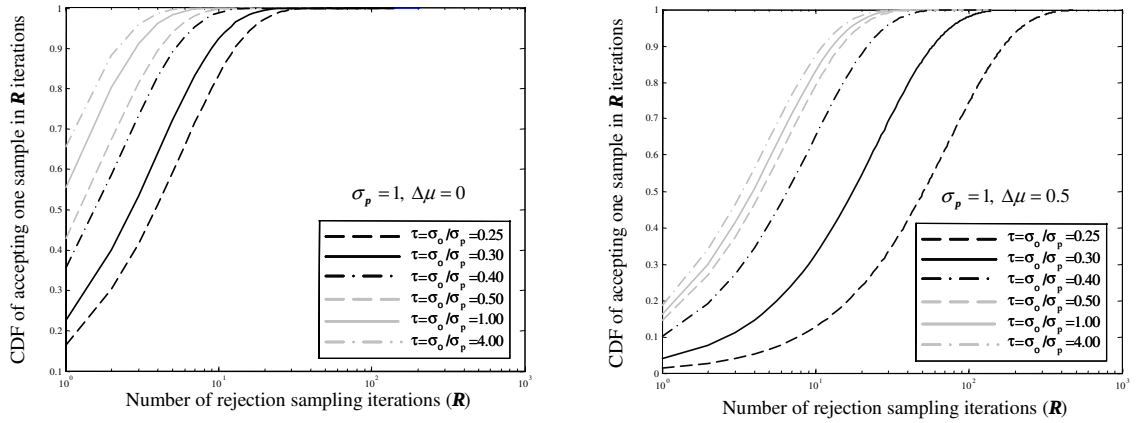


Figure 4.3: The cumulative distribution function (CDF) of the number of rejection sampling iterations (R) until the first accepted sample, modeled as a Bernoulli process and for a Bayes prior (either a real prior or that prior after applying the transition model) and an observation model normally distributed. In the image on the left, the means of both Gaussians coincide ($\Delta\mu = 0$), whereas for the case at the right hand they do not. As a consequence more trials are required in average to obtain an accepted sample in the second case. The parameter $\tau = \sigma_o/\sigma_p$ controls the relative variance of each Gaussian (σ_o and σ_p stand for the standard deviations of the observation likelihood and the prior, respectively). Observe how more trials are needed as the observation model becomes more peaked (lower τ values).

The number of Bernoulli trials required to obtain the first success, denoted by R, is known to follow a geometric probability distribution:

$$P(R) = p(1-p)^{R-1} \tag{4.4.1}$$

where p states the probability of success for each individual trial. Provided this parameter, the expected number of trials until the first success is then given by $E[R] = 1/p$, which determines the average performance of our algorithm.

Unfortunately, the parameter p cannot be computed in closed-form for a generic Bayes prior and observation models. In general, this probability will be high if the Bayes prior in the filter coincides with the observation likelihood of the last observation. To illustrate this we have derived an analytical solution for the expected value of p (refer

to Appendix C) in the specific situation of both the prior and the observation model being Gaussians. The cumulative distribution function (CDF) of R, given by:

$$CDF(R) = 1 - (1 - p)^R \tag{4.4.2}$$

is represented in Figure 4.3 for p values in two different cases: the prior and the observation likelihood being centered at the same point ($\Delta\mu = 0$, top graph), and separate ($\Delta\mu = 0.5$, bottom graph). It is clear how the first case requires fewer trials than the later for a given CDF of succeeding. For both cases we have also swept the ratio between the standard deviations of the prior (σ_p) and the observation likelihood (σ_o), which is reflected by the parameter τ: a low value indicates a "narrow" observation model, i.e. a precise sensor. The results confirm that a more precise sensor will require more trials on average, an effect that becomes stronger for a larger mismatch between the prior and the observations: observe how the curve for $\tau = 0.25$ is farther from the rest in the case of $\Delta\mu = 0.5$.

Therefore, we can state that the whole time complexity of our filtering algorithm increases as the mismatch between the Bayes prior and the last observation becomes larger. In other words, the optimal particle filter will run faster for better motion models, and slower for very precise sensors (in turn, the accepted samples will be always consistent with both motion and observation models). In principle, there is not an upper bound for the time consumed by our algorithm due to the randomness of rejection sampling, although in practice we have obtained acceptable execution times as shown in the following experiments. However, if we desire a hard bound to this time, our algorithm could be modified to account for a maximum number of rejection sampling trials, at the expense of having samples with non-equal weights and thus losing the optimality in the distribution of samples.

4.5 Experimental evaluation and discussion

4.5.1 Experimental setup

The following localization experiments consist of tracking the pose of a mobile robot equipped with a laser range finder while it is manually guided through an office environment. In this experiment we employed our mobile robot *Sancho*, shown in Figure 4.4, which is built upon a Pioneer 3DX mobile base and is equipped with a Bumblebee stereo camera, a front SICK laser scanner, a rear HOKUYO laser scanner and a laptop for autonomous performance.

Figure 4.4: Our mobile robot Sancho, employed in the experiments discussed in this chapter.

The path described by the robot and the map (built using our RBPF-based SLAM technique described in Chapter 8) of the environment are shown in Figure 4.5(a). The purpose of the first experiment is to compare the accuracy in the localization between our optimal sampling mechanism and the standard proposal distribution (the robot motion model). The resolution of the occupancy grid is 0.04m, and the non-parametric

observation model is the likelihood field proposed by Thrun *et al.* in [Thr01a, Thr05].

The accuracy of the estimation has been calculated by averaging the localization errors of all the particles at each time step, and using as the ground truth the robot poses estimated while the map was first built. To get significant results we have performed 100 executions for each sample size, ranging from just one particle up to one hundred. Note that the capability of adapting the sample size in our algorithm has been disabled in this first experiment to provide a fair comparison to a standard PF. The most interesting conclusion from the results, plotted in Figure 4.5(b), is that our optimal PF has an excellent performance starting from just one particle (an average error of roughly 0.10m), whereas a standard proposal distribution algorithm needs about 10 particles or more to avoid the filter to diverge (e.g. the average error of 6m for one particle implies that the estimated path is far from the real one!).

On the other hand, our method requires more computation time than the standard approach. For example, for 100 particles, ours takes 50.56sec. for tracking the robot along the whole path, while the standard PF takes only 9.91sec under the same computational conditions. Thus, one could argue that a standard PF with more particles would achieve a similar accuracy than our optimal PF for the same computation time. Actually, we can see in the graphs that our method always achieves a better accuracy than the standard approach, even with much fewer particles and a similar computation time. For example, our method takes the same time with 12 particles than 80 particles in the standard filter, though they achieve average errors of 7.03cm and 9.50cm, respectively (refer to Figure 4.5(b)).

We also report here a second localization experiment where the adaptive sample size capability of our algorithm is enabled. In this case, we start in the situation of global localization (or the *awakening problem*), that is, initially a large number of particles (10^4) are distributed uniformly over the whole environment. As shown in Figure 4.5(c),

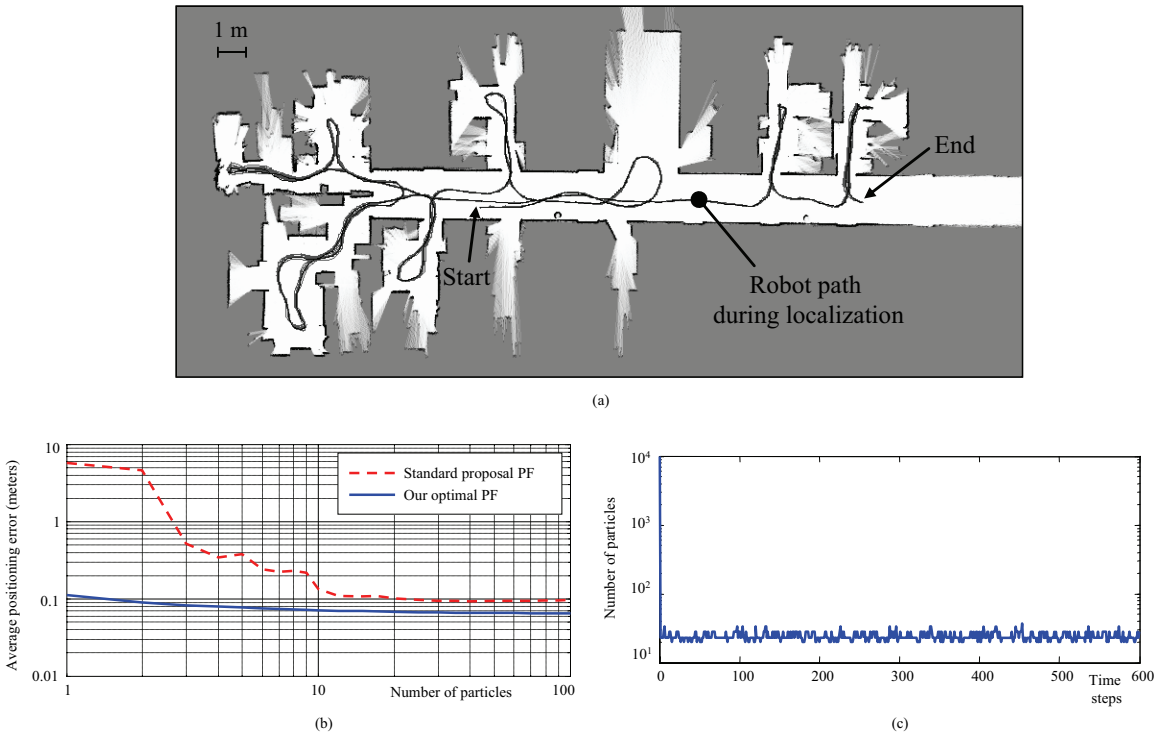


Figure 4.5: Results for localization experiments with the proposed algorithm. (a) The map used for the experiment and the ground-truth path through the environment. (b) The average errors in positioning along the entire path of the robot, using a particle filter (PF) with the standard proposal and our optimal algorithm, both for different sample sizes (results averaged over 100 repetitions). Observe how our method performs well even for just one particle, whereas the standard method diverges. (c) The adaptive number of particles for our method, when starting in a situation of global localization (10^4 uniformly distributed particles).

the sample size drastically falls in the first few iterations to the range of 20-30 particles, and it remains approximately fixed along the whole experiment. This is because there are no situations where the sensor readings become particularly ambiguous. Since our technique for achieving an adaptive number of samples is the method proposed by Fox in [Fox03], we do not provide experiments evaluating this feature here.

4.5.2 Discussion

In this chapter we have identified situations (localization and SLAM with occupancy grid maps) that require a solution based on particle filters and where optimal filtering was not directly applicable due to non-parametric observation models. We have introduced a novel algorithm that allows us to apply optimal filtering to those dynamic systems by means of simulations based on rejection sampling and an adaptive sample size. The method is able to focus the samples on the relevant areas of the state space better than previous particle filter algorithms, which has been confirmed experimentally by means of synthetic experiments and also by successfully tracking the pose of a mobile robot even with just one particle. The presented algorithm has numerous potential applications to a variety of estimation problems where the lack of a parametric model of observations prevented the usage of optimal filtering.

CHAPTER 5

A CONSENSUS-BASED OBSERVATION LIKELIHOOD MODEL FOR PRECISE SENSORS

5.1 Introduction

A fundamental component of Bayesian filtering is the observation likelihood, since it contributes the new information gathered by the sensors to the estimation. Unfortunately, the likelihood of an observation z given a robot pose x, usually denoted as $p(z|x)$, cannot be computed exactly since measurements depend on the sensor pose into the environment and also on the *actual* environment itself.

Formally, we could extend the sensor model as $p(z|x, m^\star)$, where $m^\star$ represents an exact model of (versus a pdf of) the real environment. In practice, the ground truth

$m^\star$ is unknown, thus the closest we can be to this model is to consider $p(z|x, m)$ as the sensor model, with $p(m)$ being an *estimation* of the map [1].

This approximation is assumed in all works on localization and SLAM, and it is unavoidable. Its effects (mainly an overconfidence in the estimation) can be ignored for non-accurate sensors (i.e. sonars), but they become a substantial problem for accurate ones: small discrepancies between the map and the real world lead to negligible and useless likelihood values. A workaround used in previous works consists in artificially inflating the uncertainty of the sensor model to account for the uncertainty in the map (typically up to two orders of magnitude above the actual sensor uncertainty). In order to integrate the likelihood values of individual range measurements of a sensor, conditional independence is typically assumed between them, that is, the product of the likelihoods of each range becomes the joint likelihood (as illustrated in the top graphs of Figure 5.1). The problem with this approximation becomes clear in dynamic environments, where there are significant differences between the expected and the actual measurements. In the example of Figure 5.1 this is schematically represented by two close measures and a discrepant one on the left. The effect in the fused likelihood if following the usual independence assumption is usually a high likelihood at robot poses that are actually inconsistent with *all* the measurements (refer to the top graphs of Figure 5.1).

A possible solution to this would be to detect outliers in the measurements and filter them out, as mentioned below in §5.2. However, this has the drawback of telling inliers from outliers, which is prone to decision errors. In this chapter it is proposed

[1] Strictly speaking, the sensor model $p(z|x)$ should be written through marginalization as $\int p(z|x, m)p(m)dm$ if only a pdf of the map $p(m)$ is known instead of an exact value $m = m^\star$. However, the shorter version $p(z|x, m)$ will be often used for brevity (see also Chapter 7).

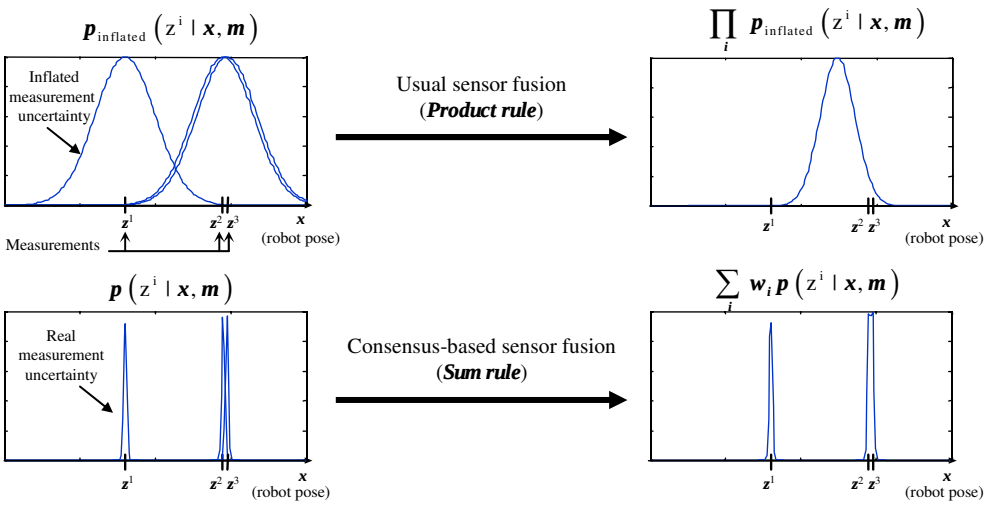


Figure 5.1: The likelihood of individual range measurements z_i from one single range scan may be contradictory in dynamic or partially known environments. The usual method to fuse them is to inflate artificially the measurement uncertainty, to assume independence, and then compute their product (graphs on the top). The result may be peaked on robot poses actually inconsistent with observations. In this chapter it is proposed a method from Consensus Theory to fuse the high-precision likelihood functions of individual measurements (bottom graphs), which leads to an accurate and robust localization.

a more appropriate approximation of the likelihood function for highly accurate sensors, concretely for laser range-finders. We consider individual likelihood values as "opinions" about the likelihood, which is actually calculated then by means of fusion methods from Consensus Theory [Gen86]. This fusion is addressed via a Linear Opinion Pool (LOP), the most simple and intuitive method from consensus fusion techniques [Ben92]. The resulting approximation of the likelihood function, that we name here Range Scan Likelihood Consensus (RSLC), allows considering the actual (very low) uncertainty of the sensor instead of some artificially inflated version of it. This approach also leads to more accurate and dependable pose estimations than existing methods, as shown later on with quantitative experiments. The advantages of using average combination rules in the presence of outliers are well known in the field of robust sensor fusion [Bor99, Kit98]. To the best of our knowledge, this approach

integrates for the first time these ideas into probabilistic robotics.

Although the new approach is applied to mobile robot localization, the sensor model should be also applicable to SLAM without modifications.

In the next section we review the previous works related to our proposal. Then, in §5.3 we set up the problem mathematically, while our new method is introduced in §5.4. We finish the chapter with experimental results that validate our approach.

5.2 Related research

Providing an observation likelihood function for accurate range scan sensors has been a challenging issue for all probabilistic approaches to localization and map building. The most physically plausible likelihood function is the beam model (BM) [Hah02, Thr05], where each range in the scan is assumed to be corrupted with zero-mean, independent identically distributed (iid) Gaussian noise. This assumption allows the following factorization:

$$p\left(z|x,m\right)=\prod_{i}p\left(z^{i}|x,m\right) \tag{5.2.1}$$

where z represents the whole scan, z_i represents individual ranges, m is the estimated map, and the expected value of each range (the mean of the corresponding Gaussian distribution $p\left(z^{i}|x,m\right)$) is computed by performing ray-tracing in the grid-map. The BM has important drawbacks in practice [Thr05]. Firstly, the resulting distribution is extremely peaked for accurate sensors, indicating the extremely small uncertainty of the sensor, thus any tiny error in the map with respect to the real world may make the distribution to diverge largely from the ground truth. Also, as a consequence of the previous drawback, if just one measurement out of the whole scan were affected

by dynamic obstacles (those non-modeled in the map) the joint distribution would become practically zero. The following solutions have been proposed in the literature to overcome these problems: (i) to inflate artificially the uncertainty in the range measurements [Thr01b], and (ii) to preprocess ranges in order to remove outliers, that is, those measurements clearly caused by dynamic obstacles [Fox99b].

An alternative to the BM is the likelihood field (LF) (also called the *end point model*), an efficient approximation that avoids the costly ray-tracing operation by taking into account the 2-d coordinates of the sensed points, and assigning likelihood values according to their nearness to correspondences in the map [Thr01a]. This model also inflates the sensor measurement uncertainty, typically up to values around one meter. In spite of its lack of a physical foundation, it has been successfully applied to localization and mapping [Gri05, Hah03]. In the context of localization, this approach can be optimized by means of precomputing the likelihood values over the entire 2-d map, becoming an extremely efficient solution.

Another alternative to BM was recently reported in [Pla07], where a more elaborate model is proposed based on Gaussian processes. In this approach, named Gaussian Beam Processes (GBP), a small uncertainty in the robot pose (most importantly, in its orientation), is taken into account to predict the uncertainty of each range in the scan, leading to much more accurate predictions than the simpler BM method.

In the two basic techniques BM and LF it is a common practice to use only a small fraction of the ranges available in the laser scans, achieving a considerable speed-up in computation times and making methods more resistant to non-modeled obstacles at the cost of suboptimal solutions [Thr05]. In the context of SLAM with Rao-Blackwellized Particle Filters [Dou00a,Hah03], an interesting alternative is proposed in [Gri05]. There the likelihood function is approximated as the Gaussian resulting from evaluating a matching function at random robot poses around a local maximum obtained from

deterministic scan matching. Its dependence on the scan matching makes it prone to local minima problems (e.g. in long corridors without salient parts).

The new method introduced in this chapter is also related to research in the field of scan matching (SM), since both observations and maps are modeled as sets of "points" (more precisely, points distributed according to 2-d Gaussians). This contrasts with the most common employment of occupancy grids [Mor85] in probabilistic localization and mapping where the robot is equipped with laser scanners [Gri05]. Most of the best-known scan matching techniques, like the Iterative Closest Point (ICP) [Bes92] or the IDC [Lu97b], aim to find the pose that achieves the optimal matching between scans. In general, these methods do not provide a measure of the uncertainty in the estimation, and hence they are not directly applicable to probabilistic robotics. There are some exceptions, like the method proposed in [Lu97a], which considers the sensor measurement uncertainty and the residuals from the least square error optimization. However, it does not take into account the uncertainty in the correspondence between points, which largely dominates the overall uncertainty. This uncertainty in the correspondences is considered in the probabilistic Iterative Correspondence (pIC) method [Mon05], but under the restrictive assumption of a normally-distributed prior of the robot pose, thus making its natural application Kalman filters.

There are still other scan matching methods [Hah02] that provide an estimation of the pose uncertainty, but they also rely on the traditional assumption of independence and product-based fusion (refer to Eq. (5.2.1)), thus they suffer from the same above-mentioned problem in dynamic environments. Unlike iterative methods [Bes92, Lu97b, Mon05], our approach does neither require distance thresholds nor is iterative, since all the uncertainty in the pose is already represented by an arbitrarily-distributed prior density.

5.3 Problem statement

The problem of mobile robot localization, including the "robot awakening" case [Fox99a], consists of estimating the robot pose x_t from all the observations $z_{1:t} = z_1, ..., z_t$ and actions $u_{1:t} = u_1, ..., u_t$ up to the current instant t, given a known map estimation $p(m)$. The distribution to estimate is then:

$$p(x_t|z_{1:t}, u_{1:t}, m) \tag{5.3.1}$$

Sequential estimation can be carried out on this by applying the Bayes rule on the most recent observation z_t, as discussed in Chapter 3, which leads to the equation (repeated here for convenience):

$$p\left(x_t|z_{1:t}, u_{1:t}, m\right) \propto \underbrace{p\left(z_t|x_t, m\right)}_{\substack{\text{Observation} \\ \text{likelihood}}} \underbrace{\int \overbrace{p\left(x_t|x_{t-1}, u_t\right)}^{\text{Motion model}} p\left(x_{t-1}|z_{1:t-1}, u_{1:t-1}, m\right) dx_{t-1}}_{\text{Prior}} \tag{5.3.2}$$

This expression indicates how to iteratively filter the robot pose distribution by means of the *observation likelihood function*, addressed in the next section.

The notation and the meaning of the involved variables, graphically represented in Figure 5.2 for clarity, is explained next.

We assume a planar robot pose, represented by $\mathbf{x}_t = [x_t\ y_t\ \phi_t]^\top$ for the time step t. Let the map m be represented as a set of fixed M points m_j, whose location uncertainty is assumed to be given by the Gaussians:

$$\begin{aligned} m &= \{m_j\}_{j=1,...,M} \\ m_j &\sim \mathcal{N}\left(\mu_m^j, \mathbf{\Sigma}_m^j\right) \end{aligned} \tag{5.3.3}$$

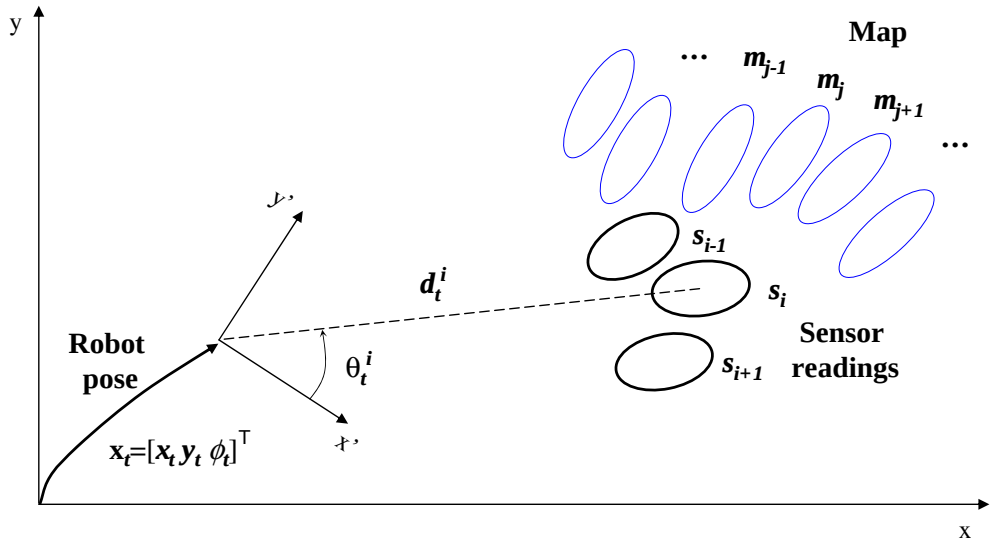


Figure 5.2: A schematic representation of the variables involved in our problem. Both the map and the observations are given by a set of normally distributed 2D points m_j. The robot pose $\mathbf{x}_t$ is used to project the sensor readings from the sensor-centric reference frame $< x', y' >$ into the global frame $< x, y >$.

Regarding the observations z_t, which represent points from a laser scan, they are described naturally, for a range scanner, in sensor-centric polar coordinates $[d_t^i\ \theta_t^i]^\top$:

$$\begin{aligned} z_t &= \left\{z_t^i\right\}_{i=1..L} \\ z_t^i &= \left[d_t^i\ \theta_t^i\right]^\top \end{aligned} \tag{5.3.4}$$

Let s_t^j be the Cartesian coordinates of the sensed point z_t^i in the global reference system $< x, y >$, once transformed from the mobile system $< x', y' >$ by (see Appendix A):

$$s_t^i = f\left(\mathbf{x}_t, z_t^i\right) = \begin{bmatrix} f_x\left(\mathbf{x}_t, d_t^i, \theta_t^i\right) \\ f_y\left(\mathbf{x}_t, d_t^i, \theta_t^i\right) \end{bmatrix} = \begin{bmatrix} x_t + d_t^i \cos\left(\phi_t + \theta_t^i\right) \\ y_t + d_t^i \sin\left(\phi_t + \theta_t^i\right) \end{bmatrix} \tag{5.3.5}$$

If we model points s_t^i as Gaussian distributions, that is,

$$s_t^i \sim \mathcal{N}\left(\mu_s^{t,i}, \mathbf{\Sigma}_s^{t,i}\right) \tag{5.3.6}$$

their means and covariance matrices can be obtained by propagating the sensor uncertainty through the linearization of the function in Eq. (5.3.5). Assuming that errors in both angles and ranges are independent and normally distributed with standard deviations σ_θ and σ_d respectively, we obtain the following parameters for the distribution of s_t^i:

$$\begin{aligned} \mu_s^{t,i} &= f\left(\mathbf{x}_t, z_t^i\right) \\ \mathbf{\Sigma}_s^{t,i} &= \mathbf{J}_f|_{z=z_t^i} \begin{bmatrix} \sigma_\theta^2 & 0 \\ 0 & \sigma_d^2 \end{bmatrix} \mathbf{J}_f^\top\big|_{z=z_t^i} \end{aligned} \tag{5.3.7}$$

where $\mathbf{J}_f$ stands for the Jacobian of the function $f(\cdot)$ in Eq. (5.3.5):

$$\mathbf{J}_f = \begin{bmatrix} \frac{\partial f_x}{\partial \theta_t^i} & \frac{\partial f_x}{\partial d_t^i} \\ \frac{\partial f_y}{\partial \theta_t^i} & \frac{\partial f_y}{\partial d_t^i} \end{bmatrix} = \begin{bmatrix} -d_t^i \sin\left(\phi_t + \theta_t^i\right) & \cos\left(\phi_t + \theta_t^i\right) \\ d_t^i \cos\left(\phi_t + \theta_t^i\right) & \sin\left(\phi_t + \theta_t^i\right) \end{bmatrix} \tag{5.3.8}$$

5.4 The range scan likelihood consensus (RSLC)

We define the RSLC as a consensus theoretic method for fusing the likelihood values of individual ranges of a scan. Consensus techniques have been employed in a variety of problems where diverse values have to be fused, e.g. for combining different classification results. We address data fusion here by means of a particular consensus method: the Linear Opinion Pool (LOP) [Ben92].

Let p be a probability density to be estimated from a set of L opinions p_i. Then, the general form for a LOP can be written as:

$$p = \sum_{i=1}^{L} \omega_i p_i$$

where ω_i are weight factors for the individual opinions. If each opinion p_i is a density function, we can assure that the result is also a density by imposing the condition:

$$\sum_{i=1}^{L} \omega_i = 1$$

For the problem we address here, p is the likelihood of a whole range scan, whereas p_i is the set of likelihood values for individual ranges in the scan. Since we cannot know in advance whether some likelihood values are more confident than others, we will simply assign an equal confidence factor to each one, leading to:

$$p\left(z_t \middle| \mathbf{x}_t, m\right) \propto \sum_{i=1}^{L} \underbrace{p\left(z_t^i \middle| \mathbf{x}_t, m\right)}_{\substack{\text{Individual} \\ \text{likelihood values}}} \tag{5.4.1}$$

which is a solution consistent with previous research on robust classification, where it is shown that this average rule for combination outperforms the classical product rule in the context of classifiers combination [Kit98].

It remains to be described how to evaluate the individual likelihood values. As previously discussed, the problem of the BM method [Thr05] is that inaccuracies in the map lead to drastic variations of the likelihood function for small displacements in the robot pose variable $\mathbf{x}_t$.

As an alternative, we propose a novel approximation for this function that, like the LF method [Thr01a], avoids the costly ray-tracing operation by considering only the Cartesian coordinates of sensed points. Concretely, we propose to approximate the likelihood of a given range measurement with the probability that the scanned point does correspond to any given map point. Put formally:

$$p\left(z_t^i \middle| \mathbf{x}_t, m\right) \propto \sum_{j=1}^{M} \mathrm{P}\left(c_{ij} \middle| \mathbf{x}_t, m_j\right) \tag{5.4.2}$$

where c_{ij} represents the correspondence between the map point m_j and the sensed point s_i, derived from z_t^i through Eq.5.3.5–5.3.7. In the computation of the correspondence probabilities we also account for the possibility of a s_i not corresponding with any particular point of the map, which is represented by $c_{i\varnothing}$. In order to compute Eq. 5.4.2 we use:

$$\mathrm{P}\left(c_{ij} \middle| \mathbf{x}_t, m\right) = \eta_i C_{ij} \tag{5.4.3}$$

Here C_{ij} is the probability density of the pair of points s_i and m_j to coincide, normalized to the range $[0, 1]$ – the role of η_i is different and is explained below. The probability density of the two points to coincide is given by the integral of both point pdf's over the whole space, which can be shown to be the evaluation of an auxiliary Gaussian at the origin (i.e. at $\mathbf{0}$), that is:

$$C_{ij} \propto \int_{-\infty}^{\infty} \underbrace{p\left(s_t^i \middle| \mathbf{x}_t, z_t^i\right)}_{\mathcal{N}\left(x;\mu_{st}^i,\mathbf{\Sigma}_{st}^i\right)} \underbrace{p\left(m_j\right)}_{\mathcal{N}\left(x;\mu_m^j,\mathbf{\Sigma}_m^j\right)} dx = \mathcal{N}\left(0; \mu_{st}^i - \mu_m^j, \mathbf{\Sigma}_{st}^i + \mathbf{\Sigma}_m^j\right) \tag{5.4.4}$$

The normalization of this quantity to the range $[0, 1]$ can be achieved by means of ignoring the normalization factors of the auxiliary Gaussian of the right hand side of Equation 5.4.4, which can be shown to lead to the expression:

$$C_{ij} = \exp\left(-\frac{1}{2} D_M^2(s_t^i, m_j)\right) \tag{5.4.5}$$

where $D_M(p, q)$ is the Mahalanobis distance between two Gaussians (as defined in §2.5).

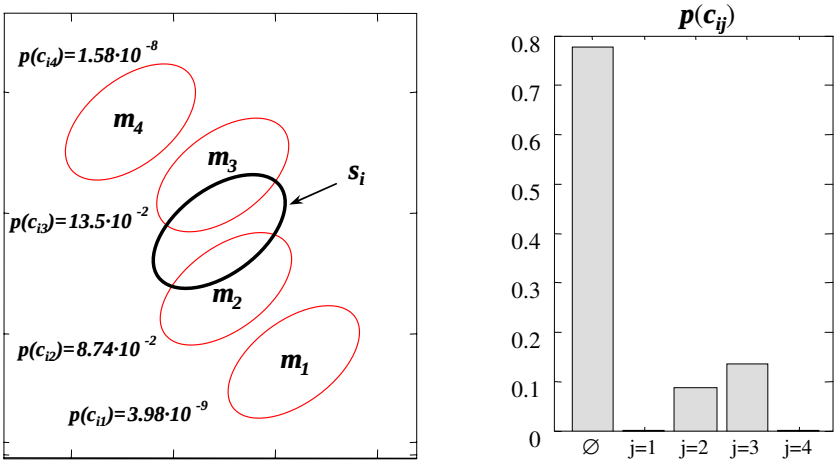


Figure 5.3: An example of how our method computes the probability of the correspondence c_{ij} between a sensed point s_i and map points m_j. A sensed point and four map points are represented in (a) among with the corresponding probability values, which are graphically represented in (b), along with the probability of the point not corresponding to any map point (the symbol $\varnothing$).

The constants η_i in Eq. 5.4.3 are computed to satisfy the law of total probability:

$$\begin{aligned} \sum_{\forall c} \mathrm{P}\left(c|\,\mathbf{x}_t, m\right) &= \\ \mathrm{P}\left(c_{i\varnothing}|\,\mathbf{x}_t, m\right) + \sum_{j=1}^{M} \mathrm{P}\left(c_{ij}|\,\mathbf{x}_t, m\right) &= 1 \end{aligned} \tag{5.4.6}$$

and we propose to model the probability of no correspondence $c_{i\varnothing}$ by means of:

$$\mathrm{P}\left(c_{i\varnothing}|\,\mathbf{x}_t, m\right) = \prod_{j=1}^{M} \left(1 - C_{ij}\right)$$

To gain an insight into these expressions, consider the example in Figure 5.3(a), where the probability of correspondence c_{ij} of a sensed point s_i is computed for a map with four points. Provided that the ellipses represent 95% confidence intervals, it is apparent that the sensed point s_i is probably a new point (it does not correspond to any point in the map). This fact is clearly reflected in Figure 5.3(b), where this

alternative receives the highest probability. Finally, according to Eq. 5.4.2, our method would assign a likelihood of $p(z^i|\mathbf{x}_t, m) = 0.2224$ to the sensed point of this example.

5.5 Experimental evaluation and discussion

Now we provide systematic comparisons between the proposed method and other two well-known approximations discussed in §5.2: the BM [Hah03] and the LF [Thr01a]. We have chosen robot localization with particle filter [Fox99a] as the framework for the tests. We also suggest the reader to view the videos available online [2] .

5.5.1 Synthetic Experiments

In the first part of the synthetic experiments, the robot pose has been estimated along a given trajectory in the environment, shown in Figure 5.4(a). The same reference map has been employed for both, (i) simulating the 361 sensor readings within the 360 degrees field-of-view of the sensor, and (ii) computing the likelihood values. By doing so, we are reproducing the situation of a perfectly known static map.

Under these conditions, we contrast the performance of particle filter-based localization for three likelihood methods: BM, LF, and our RSLC. The accuracy of the estimated pose is evaluated in two different ways:

- By computing the mean error between the ground truth (an arbitrary described by the robot in its synthetic environment) and the mean robot pose according to the particles, and
- By recovering a continuous version of the robot pose pdf from the particles by means of a Parzen window with a Gaussian kernel [Par62]), then evaluating its

[2] See: http://www.youtube.com/watch?v=atp7sYtT_dc

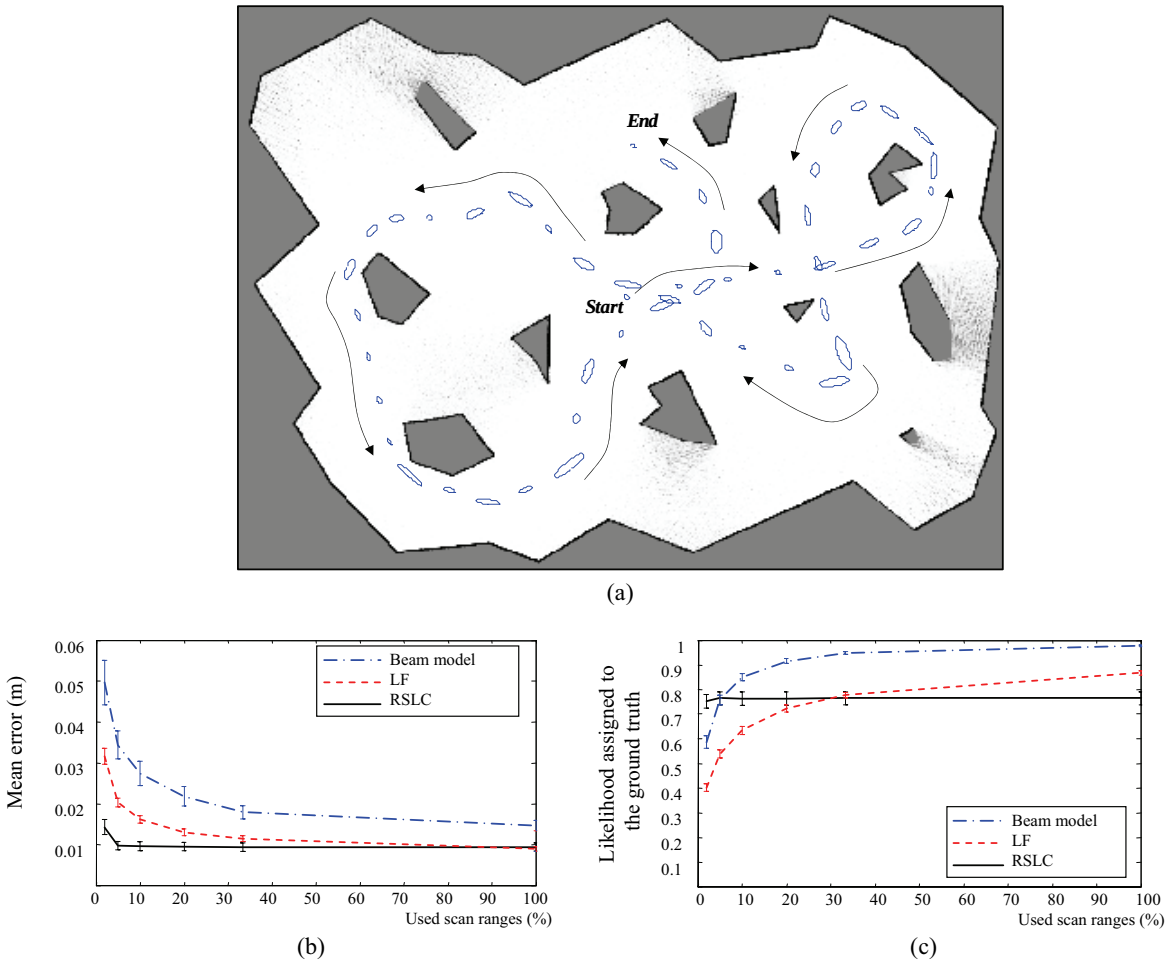


Figure 5.4: First experiments in a synthetic environment with (a) a map that perfectly models the environment. The resulting mean error from the ground truth and the likelihood assigned to the actual robot pose are shown in (b)–(c) respectively for the three methods (BM, LF, and RSLC). The graphs show the evolution of the results with the percentage of employed ranges from the scan. Confidence intervals of 68% are marked in all the graphs.

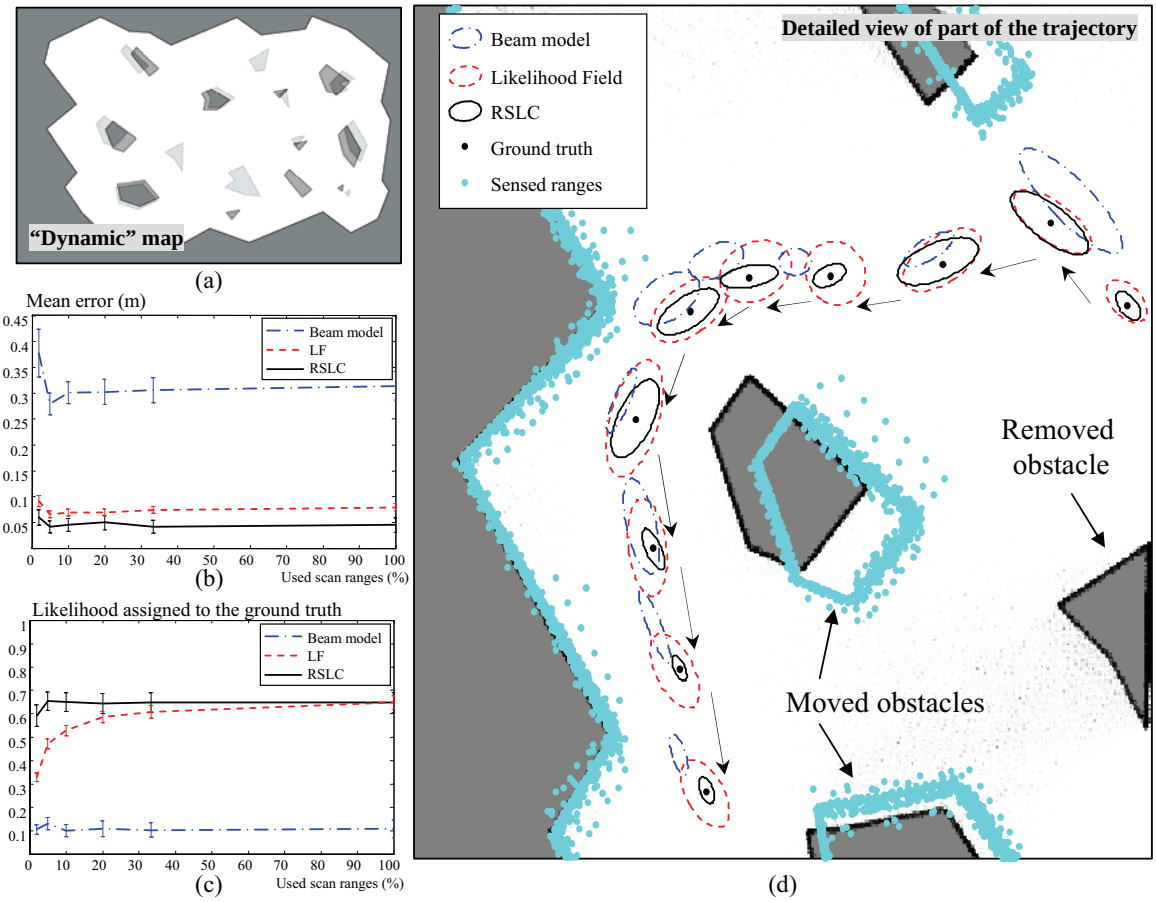


Figure 5.5: Second experiment for a synthetic, dynamic environment, emulated by using a slightly different map (a), whose results are plotted in (b)–(d), which reveal that RSLC outperforms the other methods. (e) A close look at part of the estimated trajectory, according to each method.

value at the ground truth. Notice that this method should assign a higher score to a pdf more focused on the actual robot pose.

The results are shown in Figure 5.4(b)–(c), respectively. Due to the stochastic nature of the experiments we represent the mean values and 1σ confidence intervals for each chart after executing each experiment 10 times. The ordinates of the graphs stand for the ratio of range values employed from the whole scan: the experiments have been repeated for ratios starting at 2% (7 ranges) and up to the whole scan (361 ranges). These results reveal that RSLC provides the most accurate pose estimation

(1cm mean error, approximately), even using only 2% of the ranges in the scan, while the LF method requires almost 100% of them to achieve the same accuracy. Regarding the probability assigned to the ground truth (see Figure 5.4(c)), the RSLC method is surpassed by the LF and BM: RSLC is too pessimistic in assigning likelihood values. Thus, for perfectly known environments, the estimations from BM and LF are less uncertain than the one from RSLC, thus RSLC is excessively pessimistic for this ideal situation and may lead to particles being more spread than actually necessary.

To emulate a dynamic scenario, sensor readings are simulated through the modified map shown in Figure 5.5(a), whereas the robot uses the "reference" map in Figure 5.4(a) to compute likelihood values. In the "dynamic" map some obstacles have been moved, removed or added to simulate typical problems. These experiments reveal a superior performance of RSLC: the mean error for our method, in Figure 5.5(b), is the lowest from the three methods over the whole range of ratio values. Even more significant are the results for the probability assigned to the ground truth: by comparing Figure 5.4(c) and Figure 5.5(c) it can be seen how this indicator decreases slightly for the RSLC, whereas it abruptly falls for the other methods. Please, notice that particle filters perform a resampling process that discard particles at poses with a low likelihood value, thus RSLC is the most robust method in the sense that it minimizes the probability of a correct particle to be removed. The robustness of RSLC can be better visualized in Figure 5.5(d), where the estimated path is shown according to each method. For ease of representation, the 99.7% confidence intervals are used instead of the original particles. It is clear that the estimation from RSLC is closer to the ground truth and less biased than the others.

5.5.2 Real Robot Experiment

To test the robustness of the different likelihood estimation methods against dynamic objects and discrepancies between the map and a real environment we have carried out an experiment where the robot moves throughout a dynamic cluttered room populated with people. A map of the environment was previously built using a RBPF (see Chapter 8), and then the scenario was modified by moving furniture, removing objects, etc. The results are summarized in Figure 5.6, where the particles are plotted for some instants of time together with the scanner readings projected from the weighted mean given by the particles. It can be visually verified that the estimation using RSLC provides a better alignment of the readings with the map, even in situations where most ranges do not have correspondences into the map.

It is remarkable the poor performance of the BM, whose estimation is absolutely wrong from time $t = 20sec$ on, approximately (please, observe that the sensed scans do not match the map at all). This is due to the inability of this model to cope with objects that appeared in the map but were removed afterwards. The opposite case (sensing new objects not present in the map) is typically solved by pre-processing the range scan [Fox99b]. By contrast, the RSLC does not require additional artifacts to deal with the problems derived from dynamic environments.

Figure 5.6: Results for localization in a real dynamic scenario. Snapshots on the left column show some instants of time along the robot navigation, while the other columns illustrate the evolution of the particle filter for each likelihood computation method. See the text for further discussion.

5.5.3 Discussion

In this chapter we have addressed the problem of deriving a likelihood function for highly accurate range scanners. Instead of assuming an unrealistic measurement uncertainty for each range, as most previous works do, an alternative approach has been presented where accurate likelihood model for individual ranges are fused by means of a Consensus Theoretic method.

Simulations in static, synthetic environments reveal an excellent performance of the RSLC method even when considering only a small fraction of the whole range scan (e.g. 7 range values), while other methods require significantly more measurements to achieve similar results. Furthermore, results for real dynamic environments demonstrate a qualitative improvement in the robustness of the robot pose estimation.

CHAPTER 6

FUSING PROPRIOCEPTIVE SENSORS FOR AN IMPROVED MOTION MODEL

6.1 Introduction

As discussed in Chapter 3, the Bayesian filtering approach to mobile robot localization demands the usage of probabilistic models for both the robot observations about the environment, and also for its own "actions" – typically in this context, the motor commands sent to the wheels.

Previous chapters have focused on the important topics of optimal filtering and how to model the observation likelihood function. In contrast, in this chapter we aim at obtaining an accurate probabilistic motion model by means of combining different

ego-motion sensors on the robot.

We will address this goal as an optimal estimation problem under the assumption of Gaussianity for the pdf of robot displacements, which is plausible for pose increments sampled with periods of a few seconds or less, which is indeed the typical situation in mobile robotics.

In spite of a number of proprioceptive ego-motion sensors existing for ground mobile robots [Bor96], odometers are included into virtually all commercially available ones. Actually, in most cases odometry is the only ego-motion sensor on the robot. Although other proprioceptive sensors like inertial measurement units (IMUs) may provide valuable information to the displacement estimation, they are not usually integrated into commercial robots. Our aim is to integrate different kinds of ego-motion sensors into a mathematically grounded way, concretely probabilistic Bayesian estimation, while still being a solution efficient enough to be integrated into the low-level firmware onboard a real robot. The utility of sensor fusion is revealed by noticing that different sensor weaknesses and advantages may complement to each other: typically, an odometer provides a quite precise estimation of translational movements but performs poorly when the robot turns. On the contrary, IMUs typically measure rotations more precisely than translations, due to the additional time integration required in the latter.

In the next section we describe the employed sensors. Then, we will derive the probabilistic filtering equations for the sensor fusion and introduce our physical implementation onto an onboard microcontroller. We finish presenting experiments for our design at work onboard a real robot.

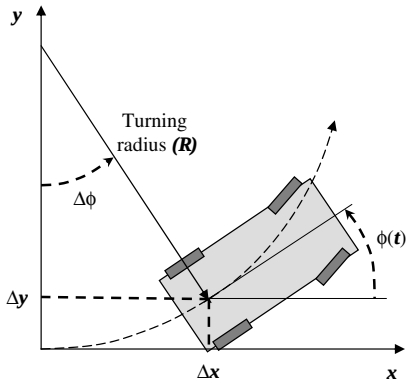


Figure 6.1: The kinematic model of a planar, differential-driven vehicle.

6.2 The sensors

In this chapter we consider a robotic wheelchair [Gon06b] equipped with two ego-motion sensors: an odometer and a gyroscope. We describe next the general kinematic model of this robot and its relation to each of the sensors. Assuming that the mobile robot moves in a planar environment, its pose is completely defined by its 2-d coordinates (x, y) and its heading angle ϕ, as sketched in Figure 6.1. The kinematic model is the well-known "tricycle model", where the robot is constrained to move in circular paths only. Provided that the robot pose is sampled at a high rate (with respect to its speed), it is reasonable to approximate the real robot path by a sequence of short circular arcs. Odometry sensors comprises two encoders[1] (one for each wheel motor), from whose readings we can compute a robot pose change $(\Delta x, \Delta y, \Delta\phi)$ for small periods of time assuming a circular path, then compose (see Appendix A) all of those arcs as a sequence in time to approximate any arbitrary path the robot followed.

On the other hand, a gyroscope is an inertial sensor which measures the instantaneous change rate of the robot orientation $\phi(t)$, that is, the yaw-rate $\frac{d\phi(t)}{dt}$. Please refer

[1] In the case of our robotic wheelchair SENA, the model of the encoders is BHK 06.05A-1024-12-5.

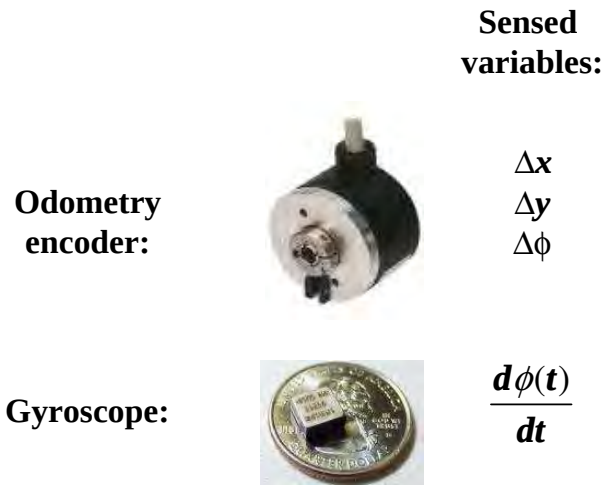


Figure 6.2: The proprioceptive sensors employed in this chapter.

to Figure 6.2 for an illustration of how these variables are related to the robot kinematics. Since each sensor has its own error sources and they measure different variables from the kinematic model, their combination into a single, optimal estimation is not straightforward. How to achieve this is the topic of the next section.

6.3 Proprioceptive sensor fusion

Here we employ an EKF [Jul97] for fusing the readings from heterogeneous proprioceptive sensors. This filter is an iterative Bayesian filter that represents, for each instant of time, a probability distribution for the system state by means of multivariate Gaussians, that is, a mean value and a covariance matrix. This distribution is modified according to *actions* (prediction step) and then is corrected according to the sensors measurements or *observations* (update step). Probabilistic models are required for both the evolution of the system and for the sensors. In the following we present the complete design of an EKF filter for the problem of tracking the pose of a mobile robot equipped with proprioceptive ego-motion sensors only. We will take odometry readings as the action of the robot since they represent what the motors have done[2], whereas gyroscope readings are considered observations of the system state because it is completely passive.

Let $\mathbf{x}_k$ be the state of our system at the discrete time-step k:

$$\mathbf{x}_k = \begin{pmatrix} x_k \\ y_k \\ \phi_k \\ \phi_{k-1} \end{pmatrix} \tag{6.3.1}$$

where the memory term ϕ_{k-1} stands for the robot orientation at the last time step. If we define Δt as the filter sampling period, the last memory term allows us to approximate the robot angular velocity ω_k as:

$$\omega_k \approx \frac{\phi_k - \phi_{k-1}}{\Delta t} \tag{6.3.2}$$

[2]This will always be more accurate than the actual commands sent out to the robot motors.

Let the estimation at time step $k-1$ be given by the normal distribution

$$\mathcal{N}(x_{k-1}; \hat{\mathbf{x}}_{k-1}, \mathbf{P}_{k-1})$$

with $\hat{\mathbf{x}}_{k-1}$ and $\mathbf{P}_{k-1}$ being the mean and the covariance matrix, respectively. Then, the prediction step of the EKF filter reads:

$$\begin{aligned} \hat{\mathbf{x}}_k^- &= f\left(\hat{\mathbf{x}}_{k-1}, u_k\right) && (6.3.3) \\ \mathbf{P}_k^- &= \begin{pmatrix} \nabla_{\mathbf{x}_k} f & \nabla_{u_k} f \end{pmatrix} \begin{pmatrix} \mathbf{P}_{k-1} & 0 \\ 0 & \mathbf{C}_{u_k} \end{pmatrix} \begin{pmatrix} \nabla_{\mathbf{x}_k} f^\top \\ \nabla_{u_k} f^\top \end{pmatrix} \\ &= \nabla_{\mathbf{x}_k} f \mathbf{P}_{k-1} \nabla_{\mathbf{x}_k} f^\top + \nabla_{u_k} f \mathbf{C}_{u_k} \nabla_{u_k} f^\top && (6.3.4) \end{aligned}$$

with $f(\cdot)$ being the transition function of the system, and u_k the action performed at time step k whose covariance matrix is $\mathbf{C}_{uk}$. The minus sign in the superscript of Eq. (6.3.3) means that the estimation is the prior in the Bayesian filter, that is, sensor observations have not being incorporated into the estimation yet. Since odometry readings are considered as the robot actions, we have $u_k = (\Delta x_k, \Delta y_k, \Delta\phi_k)^\top$, and, according to the robot kinematic model, the transition function becomes:

$$f\left(\hat{\mathbf{x}}_{k-1}, u_k\right) = \begin{pmatrix} x_k \\ y_k \\ \phi_k \\ \phi_{k-1} \end{pmatrix} = \begin{pmatrix} x_{k-1} + \Delta x_k \cos\phi_{k-1} - \Delta y_k \sin\phi_{k-1} \\ y_{k-1} + \Delta x_k \sin\phi_{k-1} + \Delta y_k \cos\phi_{k-1} \\ \phi_{k-1} + \Delta\phi_k \\ \phi_{k-1} \end{pmatrix} \quad (6.3.5)$$

It is straightforward to derive from Eq. (6.3.5) the Jacobian matrixes $\nabla_{\mathbf{x}_k} f$ and $\nabla_{\mathbf{x}_{u_k}} f$ required to evaluate Eq. (6.3.3), hence they are omitted here. Next, the update step is performed in the iterative filter:

$$\begin{aligned} \hat{\mathbf{x}}_k &= \hat{\mathbf{x}}_k^- + \mathbf{K}_k \tilde{y}_k \\ \mathbf{P}_k &= \left(\mathbf{I}_4 - \mathbf{K}_k \mathbf{H}_k\right) \mathbf{P}_k^- \end{aligned} \quad (6.3.6)$$

where $\mathbf{I}_4$ is the 4×4 unit matrix, and:

$$\begin{aligned}
\tilde{y}_k &= z_k - h\left(\hat{\mathbf{x}}_k^-\right) \\
\mathbf{S}_k &= \begin{pmatrix} \nabla_{\mathbf{x}_k} h & \nabla_{S_A} h & \nabla_n h \end{pmatrix} \begin{pmatrix} \mathbf{P}_k^- & 0 & 0 \\ 0 & \sigma_{S_A}^2 & 0 \\ 0 & 0 & \sigma_n^2 \end{pmatrix} \begin{pmatrix} \nabla_{\mathbf{x}_k} h^\top \\ \nabla_{S_A} h^\top \\ \nabla_n h^\top \end{pmatrix} \\
\mathbf{K}_k &= \mathbf{P}_k^- \mathbf{H}_k^\top \mathbf{S}_k^{-1}
\end{aligned} \tag{6.3.7}$$

Basically, this step predicts the expected sensor (gyroscope) outcome through the sensor model $h(\cdot)$, its uncertainty (covariance matrix $\mathbf{S_k}$), and also fuse this information with the prior estimation. The gyroscope sensor model could be defined as:

$$h\left(\hat{\mathbf{x}}_k\right) = \hat{\omega}_k = \frac{\hat{\phi}_k - \hat{\phi}_{k-1}}{\Delta t} \tag{6.3.8}$$

However, the actual sensor readings are analog voltage values, thus we must consider two uncertainty sources in the gyroscope readings z_k: (i) electrical noise, modeled as additive white Gaussian noise (AWGN) with variance σ_n^2, and (ii) uncertainty in the actual sensor sensitivity, i.e. the volts to deg/s ratio. To integrate the effects of both error sources into the EKF we rewrite Eq. (6.3.8) to account for S_A and S_N, the actual (unknown) and nominal sensor sensitivity, respectively, and for the noise n_k:

$$h\left(\hat{\mathbf{x}}_k\right) = \left(\frac{\hat{\phi}_k - \hat{\phi}_{k-1}}{\Delta t} S_A + n_k \right) \frac{1}{S_N} \tag{6.3.9}$$

According to the data available in the manufacturer supplied datasheet for our gyroscope, an ADXRS401[3], it seems that the sensitivity of the device, taken as the outcome of a random variable for each chip specimen, approximately follows a Gaussian distribution. Therefore, it makes sense to estimate the covariance matrix S_k by

[3]Online datasheet is available at http://www.analog.com/

linearization of the sensor model in Eq. (6.3.9), as shown in Eq. (6.3.7). There, the three Jacobians of the function $h(\cdot)$, $\nabla_{\mathbf{x}_k} h$, $\nabla_{S_A} h$, and $\nabla_{S_n} h$, describe how the uncertainty in the system state x_k, the sensitivity S_A, and the electrical noise n_k, correspondingly, are reflected in readings from the gyroscope. After each iteration of the filter we obtain the updated optimal estimation, disregarding linearization errors.

6.4 Implementation

Next we discuss how the theoretical filter described above has been integrated into the low-level firmware of a mobile robot. The system has been designed to work in a timely fashion, under hard real-time requirements. At a working rate of 100Hz, the system collects readings from encoders (odometry) and the gyroscope, performs the required preprocessing of signals, and executes an iteration of the EKF as detailed in §6.3. A logical overview of the system is provided in Figure 6.3. The signal conditioning stage is required since our gyroscope (ADXRS401) presents a nonratiometric analog output, while analog-digital converters (ADCs) are ratiometric[4]. Therefore, the resulting readings are highly sensitive to electrical noise coupled to the power supply, which is a major issue on a mobile robot, where motors produce large noise while in operation. To solve this problem, both the sensor output voltage and its 2.5V constant voltage reference are converted through ADCs. The purpose of the signal conditioning stage (please, refer to Figure 6.3) is two-fold: (i) to scale the sensor readings according to the constant voltage reference; and (ii) to remove the sensor offset, that is, to precisely determinate the voltage corresponding to a null yaw-rate.

The offset voltage can be easily estimated by averaging over a time sliding window

[4]Ratiometric means that the output is proportional to the power supply voltage and nonratiometric means it is independent of that voltage.

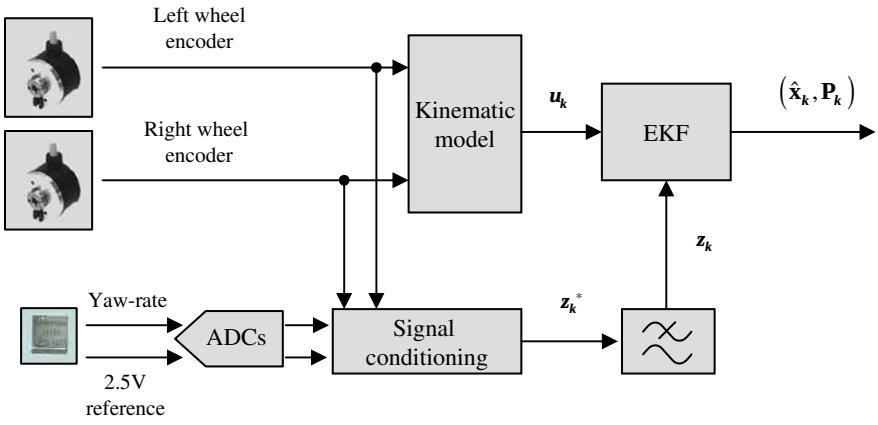


Figure 6.3: A schematic view of our implementation of the ego-motion estimation system.

when the robot is very likely at rest, e.g. when odometry does not detect motion for a few seconds.

After removing the offset from the gyroscope signal, it still contains high-frequency electrical noise, which does not carry information about the mechanical system. Since the analog circuitry of the gyroscope has been set up for a measuring bandwidth of 5Hz, we can disregard the signal components at higher frequencies as undesirable noise. In our system, the noise is filtered out through a Finite Impulse Response (FIR) implementation of a fourth order elliptic low-pass filter, with a nominal pass-band ripple of 0.1dB, 30dB stopband attenuation, and a cut-off frequency of 5Hz. An example of the signal before and after noise filtering is illustrated in Figure 6.4.

The whole system has been implemented on an ATMEGA128 [5]: a low-cost, 8-bit RISC microcontroller that runs up to 16MHz. In Figure 6.5 it is shown the prototype developed in this work, which includes two Micro-Electro-Mechanical Systems (MEMS): the already introduced gyroscope ADXRS401, and a two-axis accelerometer ADXL203, which can be used to detect the gravity vector, i.e. tilt sensing, although

[5]Refer to the manufacturer website: http://www.atmel.com/

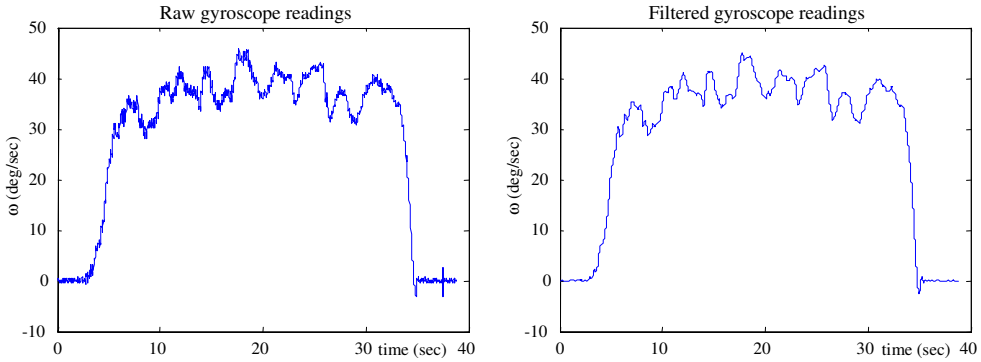


Figure 6.4: The real signal gathered from the gyroscope, (a) unprocessed, and (b) after filtering out the noise.

such issue is not addressed here. The system runs autonomously and periodically reports the EKF estimation results to a host-PC via a high-speed USB connection. This prototype has been designed with the aim of minimizing costs and weight.

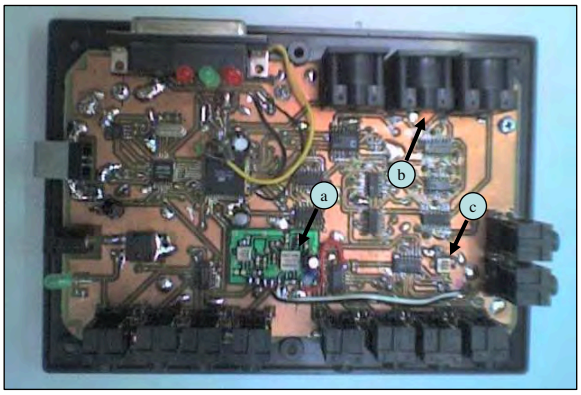


Figure 6.5: The prototype used in this work as a test bed for sensor fusion. They have been highlighted: (a) the MEMS gyroscope, (b) odometer encoders input, and (c) accelerometers for tilt sensing.

6.5 Experimental evaluation and discussion

Two comparative experiments are reported next where the robot follows two different paths while its pose is estimated simultaneously from both odometry only, and from our sensor fusion system. The actual final robot pose for each trajectory has been determined by a highly-precise laser range scan matching algorithm [Bes92], which we will consider the ground-truth for comparison purposes. The two different paths consist of moving the robot on a twisty forward, and a spinning trajectory, respectively. Results for the first experiment are summarized in Figure 6.6(a)–(d). It is noticeable the reduction in the final pose uncertainty, both in the robot position and its orientation, for the case of sensor fusion with respect to the odometry estimation only. This is numerically confirmed by the values of the covariance matrix determinant: $1.5461 \cdot 10^{-11}$ from odometry only, and $2.0429 \cdot 10^{-13}$ for sensor fusion.

In the case of the spinning trajectory, shown in Figure 6.6, the robot turns three times, which causes the odometry-only estimation to completely lose the robot orientation. The incorporation of yaw-rate information in the estimation provides an impressive qualitative improvement here: the determinant of the covariance matrix,

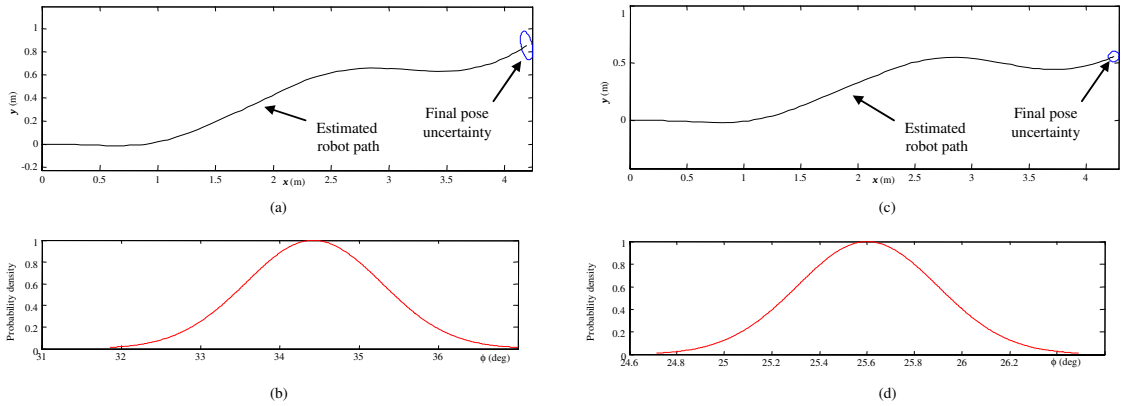


Figure 6.6: First part of the experimental results from our method for sensor fusion. The robot follows here a twisty forward trajectory. (a)–(b) Final estimated pose using odometry information only, and (c)–(d) using both odometry and the readings from the gyroscope.

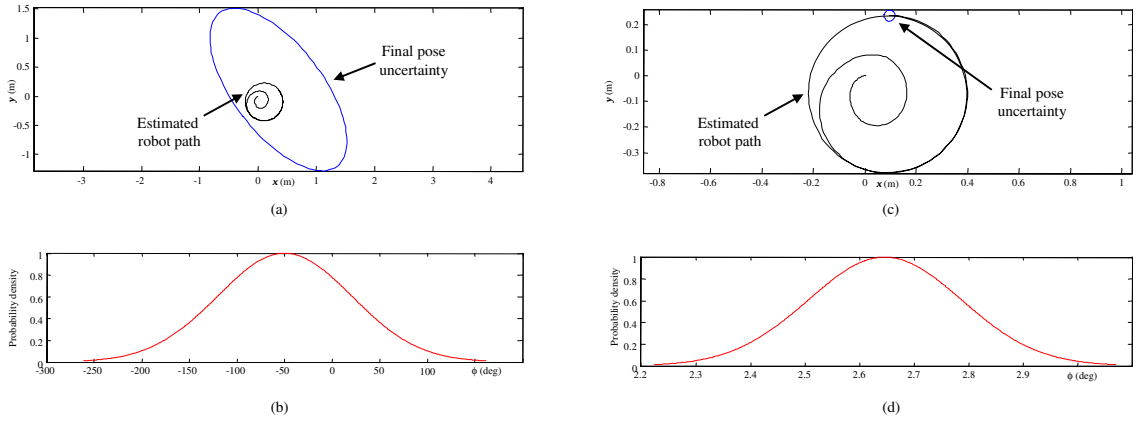


Figure 6.7: Second part of the experimental results from our method for sensor fusion. The robot follows here a spinning trajectory. (a)–(b) Final estimated pose using odometry information only, and (c)–(d) using both odometry and the readings from the gyroscope.

	Experiment I: Forward			Experiment II: Spinning		
	x	y	ϕ	x	y	ϕ
Odometry	4.194m	0.849m	34.42º	0.350m	0.114m	-49.55º
Sensor fusion	4.187m	0.934m	25.32º	0.096m	0.233m	2.65º
Ground truth	**4.169m**	**1.031m**	**25.80º**	**0.072m**	**0.282m**	**2.50º**

Figure 6.8: Summary of the experimental results: final estimated pose from each method.

$3.9538 \cdot 10^{-3}$ for the odometry-only estimation, becomes $9.1411 \cdot 10^{-15}$ for the sensor fusion case. As expected, the absolute positioning errors (relative to the ground-truth) are also reduced by the fusion of the two sensors, as summarized in Figure 6.8.

To sum up, in this chapter we have presented the problem of proprioceptive sensor fusion for ego-motion estimation for a mobile robot. An efficient solution has been proposed for the case of a robot equipped with odometry and a gyroscope, which has been implemented in the low-level firmware of a real robot and runs in real-time. Experimental results demonstrate that the sensor fusion system provides a major improvement in the quality of the robot pose estimation.

Part II

Simultaneous Localization and Mapping (SLAM)

CHAPTER 7

OVERVIEW

Automated mapping of unknown environments is one of the fundamental problems to be solved for achieving truly autonomous mobile robots. The hardness of this task follows from the fact that a precise map can only be obtained from a well-localized robot, but in turn the quality of the robot pose estimation depends on the map accuracy: this is the Simultaneous Localization and Mapping (SLAM) problem.

Although we can find pure *topological* approaches to SLAM [Ran06], in the following chapters the focus is on pure *metric* methods only. In the last years, methods based on Estimation Theory have dominated the research in this field.

In this paradigm, a sequence of robot actions $u^t = \{u_1, .., u_t\}$ and observations $z^t = \{z_1, ..., z_t\}$ are used to infer the probability distributions of robot poses $x^t = \{x_1, .., x_t\}$ and of the map m. Note the lack of subscript for the map m, which means that we assume a static world model. The corresponding DBN reflecting the causal relationships between the variables is displayed in Figure 7.1.

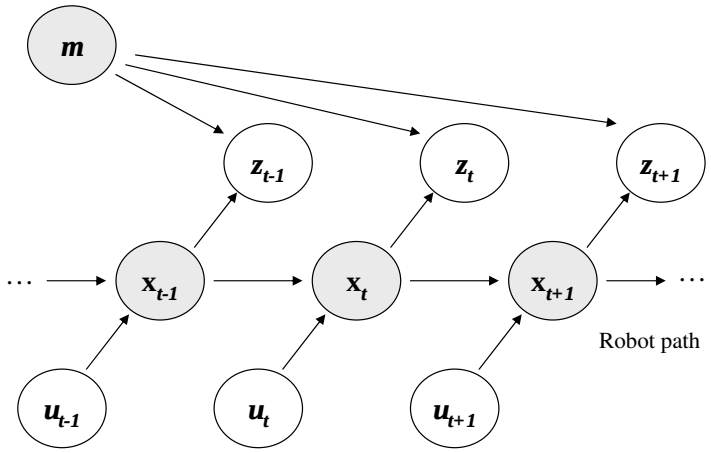


Figure 7.1: The dynamic Bayesian network for mobile robot SLAM, where robot poses x_t and the map m are hidden variables (represented as shaded nodes) to be estimated from actions u_t and sensor observations z_t.

Within probabilistic SLAM based on Bayesian filtering, there exist two different approaches for estimating the robot pose and map joint distribution [Thr05]:

- **Full SLAM:** The map and the complete history of robot poses are estimated at each instant t, that is,

$$p(x^t, m|u^t, z^t).$$

- **Online SLAM:** Only the latest robot pose and the map are estimated for each time step t, that is,

$$p(x_t, m|u^t, z^t).$$

All the methods presented in subsequent chapters will focus on full SLAM, but a probabilistic derivation of the filtering equations for both methods is presented next for completeness.

In the case of online SLAM, we have

$$\underbrace{p(x_t, m|z^t, u^t)}_{\text{Posterior for } t} \qquad \text{Bayes rule on } z_t \propto$$

$$p(z_t|x_t, m, z^{t-1}, u^t)p(x_t, m|z^{t-1}, u^t) \qquad z_t \perp\!\!\!\perp z^{t-1}, u^t \mid x_t, m =$$

$$\underbrace{p(z_t|x_t, m)}_{\text{Observation model}} \; p(x_t, m|z^{t-1}, u^t) \qquad \text{Law of total probability on } x_{t-1} \text{ to obtain recursivity} =$$

$$p(z_t|x_t, m) \int_{-\infty}^{\infty} p(x_t, m|z^{t-1}, u^t, x_{t-1}) p(x_{t-1}|z^{t-1}, u^t) dx_{t-1} \qquad x_t \perp\!\!\!\perp m \mid x_{t-1}, u_t =$$

$$p(z_t|x_t, m) \int_{-\infty}^{\infty} \underbrace{p(x_t|z^{t-1}, u^t, x_{t-1}) p(m|z^{t-1}, u^t, x_{t-1})}_{\text{Factored due to conditional independence}} p(x_{t-1}|z^{t-1}, u^t) dx_{t-1} \qquad =$$

$$p(z_t|x_t, m) \int_{-\infty}^{\infty} p(x_t|z^{t-1}, u^t, x_{t-1}) \underbrace{p(m|z^{t-1}, u^t, x_{t-1}) p(x_{t-1}|z^{t-1}, u^t)}_{\text{These terms can be joined}} dx_{t-1} \qquad p(a|b)p(b) = p(a, b) =$$

$$p(z_t|x_t, m) \int_{-\infty}^{\infty} p(x_t|z^{t-1}, u^t, x_{t-1}) p(x_{t-1}, m|z^{t-1}, u^t) dx_{t-1} \qquad u_t \perp\!\!\!\perp x_{t-1}, m \mid \varnothing =$$

$$p(z_t|x_t, m) \int_{-\infty}^{\infty} p(x_t|z^{t-1}, u^t, x_{t-1}) \underbrace{p(x_{t-1}, m|z^{t-1}, u^{t-1})}_{\text{Posterior for } t-1} dx_{t-1} \qquad x_t \perp\!\!\!\perp z^{t-1}, u^{t-1} \mid u_t, x_{t-1} =$$

$$p(z_t|x_t, m) \int_{-\infty}^{\infty} \underbrace{p(x_t|u_t, x_{t-1})}_{\text{Transition model}} p(x_{t-1}, m|z^{t-1}, u^{t-1}) dx_{t-1}$$

In principle, the filtering expression obtained for online SLAM can be implemented by means of both Kalman-like filters (EKF, IKF,...) and particle filters. Nevertheless, the characteristics of the problem make the former more suitable and thus they are widely employed in this context [Dis01, Por06]. The reason for this advantage of Kalman-like filters is that they model the robot pose and the map as a single state vector, including the cross-covariances between all its elements, and therefore making

unnecessary to keep other robot poses except the latest one [1]. For example, in a loop closure the whole robot path suffers a correction, but, as long as correlations are known between all the landmarks in the map, the observation of the latest landmarks automatically modifies all the previous ones without the need of recomputing the corrected robot path.

EKF-based SLAM is probably the most oft-used solution to SLAM found in the literature, sometimes even becoming a synonymous with SLAM [Bai06a]. However, in spite of its success for small or mid-sized landmark maps, EKF-based SLAM suffers from severe limitations due to errors introduced by linearization and its poor scalability, a consequence of the need to keep non-sparse covariance matrices with sizes in the order of $N \times N$ for maps of N elements.

Full SLAM was proposed as a response to these limitations, since the estimation of the whole robot path makes the elements in the map independent [2], and therefore the covariance matrices of the whole map (or its equivalent for maps not based on landmarks) become diagonal block matrices, which greatly simplifies the overall estimation problem.

On the other hand, full SLAM carries the cost of parametric filters not being applicable anymore to the joint distribution [Hah03], leaving particle filters as the unique feasible solution. The idea on which full SLAM relies is Rao-Blackwellization [Dou00a], where the factorization of the full joint

[1] This follows from the robot poses being a Markov process, as can be verified in the graphical model in Figure 7.1.

[2] Actually, *conditionally* independent between them, given the robot path.

$$
\begin{aligned}
p(x^t, m|z^t, u^t) &\overset{p(a,b)=p(b)p(a|b)}{=} \\
p(x^t|z^t, u^t)p(m|x^t, z^t, u^t) &\overset{m \perp\!\!\!\perp u^t \mid x^t, z^t}{=} \\
\underbrace{p(x^t|z^t, u^t)}_{\text{Robot path}} \underbrace{p(m|x^t, z^t)}_{\text{Map}} &
\end{aligned}
\tag{7.0.1}
$$

is introduced with the aim of estimating the robot path x^t with one particle filter (first term in Eq. (7.0.1)), then update the map m independently (second term). This approach will be revisited and discussed in more detail along subsequent chapters.

Regarding the second term in Eq. 7.0.1, the map density $p(m|x^t, z^t)$ has closed-form solutions for occupancy grid maps [Thr03] and landmark maps (in which case, parametric filters can indeed be employed [Mon03a]). The interesting estimation problem arises then in the first term: the robot path. Operating on that term, we obtain:

$$
\underbrace{p(x^t|z^t, u^t)}_{\text{Posterior of the path for } t} \overset{\text{Bayes rule on } z_t}{\propto} p(z_t|x^t, z^{t-1}, u^t)p(x^t|z^{t-1}, u^t) \overset{\substack{\text{Law of total probability on } x^{t-1} \\ \text{to obtain recursivity}}}{=}
$$

$$
p(z_t|x^t, z^{t-1}, u^t) \cdot \int \cdots \int_{-\infty}^{\infty} p(x_t, x^{t-1}|z^{t-1}, u^t, x^{t-1})p(x^{t-1}|z^{t-1}, u^t)dx^{t-1}
$$

$$
\overset{x^{t-1} \perp\!\!\!\perp u_t \mid \varnothing}{=}
$$

$$
p(z_t|x^t, z^{t-1}, u^t) \cdot \int \cdots \int_{-\infty}^{\infty} p(x_t, x^{t-1}|z^{t-1}, u^t, x^{t-1}) \underbrace{p(x^{t-1}|z^{t-1}, u^{t-1})}_{\text{Posterior of the path for } t-1} dx^{t-1}
$$

$$
\overset{p(a,b|b)=p(a|b)}{=}
$$

$$
p(z_t|x^t, z^{t-1}, u^t) \cdot \int \cdots \int_{-\infty}^{\infty} p(x_t|z^{t-1}, u^t, x^{t-1})p(x^{t-1}|z^{t-1}, u^{t-1})dx^{t-1}
$$

$$\overset{x_t \perp\!\!\!\perp x^{t-2}, u^{t-1}, z^{t-1} | x_{t-1}, u_t}{=}$$

$$p(z_t|x^t, z^{t-1}, u^t) \cdot \int \cdots \int_{-\infty}^{\infty} \underbrace{p(x_t|x_{t-1}, u_t)}_{\text{Transition model}} p(x^{t-1}|z^{t-1}, u^{t-1}) dx^{t-1}$$

$$\overset{\text{Law of total probability on } m}{=}$$

$$\int_{-\infty}^{\infty} p(z_t|x^t, z^{t-1}, u^t, m) p(m|x^t, z^{t-1}, u^t) dm \cdot \underbrace{\int \cdots \int_{-\infty}^{\infty} p(x_t|x_{t-1}, u_t) p(x^{t-1}|z^{t-1}, u^{t-1}) dx^{t-1}}_{\text{Bayes prior of the robot pose for } t}$$

$$\overset{z_t \perp\!\!\!\perp x^{t-1}, z^{t-1}, u^t \mid x_t, m}{=}$$

$$\int_{-\infty}^{\infty} \underbrace{p(z_t|x_t, m)}_{\text{Observation model}} p(m|x^t, z^{t-1}, u^t) dm \cdot \int \cdots \int_{-\infty}^{\infty} p(x_t|x_{t-1}, u_t) p(x^{t-1}|z^{t-1}, u^{t-1}) dx^{t-1}$$

$$\overset{m \perp\!\!\!\perp x_t, u^t \mid x^{t-1}, z^{t-1}}{=}$$

$$\int_{-\infty}^{\infty} p(z_t|x_t, m) \underbrace{p(m|x^{t-1}, z^{t-1})}_{\text{Map estimate for } t-1} dm \cdot \int \cdots \int_{-\infty}^{\infty} p(x_t|x_{t-1}, u_t) p(x^{t-1}|z^{t-1}, u^{t-1}) dx^{t-1}$$

where the first integral in the last expression is the sensor model, $p(z_t|x_t, m)$, integrated in order to account for the uncertainty in the map m.

When full SLAM is implemented with particle filters, the first integral in that expression should be performed over the robot path hypothesis $x^{[i],t-1}$, corresponding to some given particle index i. In order to simplify the notation of this integral, which appears quite often in particle filter-based SLAM, it will be denoted as conditioned to $m^{[i]}$, that is,

$$p(z_t|x^t, m^{[i]}) \doteq \int_{-\infty}^{\infty} p(z_t|x_t, m) p(m|x^{[i],t-1}, z^{t-1}) dm \tag{7.0.2}$$

In spite of the utility of this notation, sometimes employed in the literature, it can be considered an abuse of notation and be misleading, since the meaning of $m^{[i]}$ is not the same that $x^{[i],t}$, which represents an *exact value* hypothesis (not a pdf) of

the robot path. Hence, writing $p(\cdot|x^{[i],t})$ makes sense whereas $p(\cdot|m^{[i]})$ does not, unless interpreted as in Eq. 7.0.2.

At this point, it is worth mentioning a new concept in SLAM that did not appear in probabilistic localization, which is related to the term:

$$p(z_t|x_t, m) \tag{7.0.3}$$

In both the present chapter and in Chapter 3, the term above has been referred to as the sensor (or observation) *likelihood function*. The reason for that is that the pdf was considered as a function where the "free" variable was the observation z_t, whereas both the robot pose x_t and the map m remained fixed.

Nevertheless, we will arrive at situations where the "free" variable is the, now unknown, map m. In those cases the same term will be called the *inverse sensor model*, following the conventions found in the literature [Thr05]. Although conceptually related, in general the inverse sensor model and the observation likelihood function will be implemented differently.

The content of the rest of this part is structured as follows. Firstly, Chapter 8 will extend the optimal filtering algorithm already presented for localization in such a way it can also be applied to SLAM. Next, the problem of SLAM with range-only sensors will be discussed in Chapter 9 and an efficient probabilistic solution presented. Finally, we also address the problem of measuring the uncertainty in SLAM in Chapter 10, including applications to robot active exploration.

CHAPTER 8

OPTIMAL PARTICLE FILTERING IN SLAM

8.1 Introduction

In this first chapter in the part devoted to SLAM, we revisit the optimal filtering algorithm presented in Chapter 4 in order to introduce the required modifications for its application to SLAM.

As mentioned in [Liu98], sequential Bayesian estimation typically considers state spaces with a dimensionality that either increases or remains constant with time. The latter, which includes the problem of mobile robot localization, can be perfectly managed by the optimal algorithm derived in Chapter 4. However, there are other problems whose dimensionality increases over time, such as the previously explained *full SLAM* (see [Thr05] and Chapter 7). In these cases, the resampling that the filtering algorithm

performs at every step may lead to a loss of information in some dimensions, e.g. the past poses of the robot. In SLAM this means that the diversity of particles for representing the robot path will be lost very quickly. This is a common problem of all particle filters, and can be alleviated by introducing *selective* resampling steps [Liu98, Gri07a], as explained in the next section.

Apart from the improvements of the new method that were mentioned in Chapter 4, this is the first time, to the best of our knowledge, that the number of particles is automatically adapted to the uncertainty present at each time step in the context of SLAM. This implies that memory and computational requirements are self-adjusted during the map building process (e.g. after closing a long loop the number of samples is largely reduced).

In the next section we explain how to integrate selective resampling into the original algorithm presented in Chapter 4. Then, in §8.3 we discus the expected evolution with time of the overall number of samples of our method, and we finish with experimental results for different real datasets.

8.2 The optimal PF with selective resampling

The presented approach is based on delaying resampling in the algorithm presented in Chapter 4 as much as possible in order to avoid the loss of diversity. This can be achieved by monitoring a measure of diversity such as the effective sample size (ESS) (see §2.8 or [Liu96]).

Thus, at each iteration we compute the ESS and, only if it is below a given threshold (typically 0.5 times the number of particles), the particles are resampled. In that case the procedure will be exactly as described in the filtering method described in Algorithm 2.

Otherwise, particles are not resampled, and their weights must be multiplied by Eq. (4.3.7). Since that expression determined the likelihood of each sample to be selected in the resampling (which has not been performed), we are integrating the particle "probability of surviving the resampling" directly into its importance weight.

In this way, the modification of weights acts as a replacement of the actual resampling. This is the crucial operation that assures the mathematical validity of the importance sampling representation at those steps when resampling does not take place.

For the sake of clarity, the modified filter is described in Algorithm 2, which can be contrasted to the original method (without selective resampling) shown in Algorithm 1 (§4.3.2). Notice that the main difference is the management of the particle weights, which were discarded in the former algorithm since resampling was performed at all iterations and thus all the particles always had the same weight.

In principle, the new method may seem very similar to standard SIR filters for the case of not resampling. However, the use of a variable number of particles in our algorithm makes more complex the method to draw particle indexes (the i in Algorithm 2) since it must be assured that all particles have the same probability of passing to the next iteration if resampling is not performed. In a traditional filter with a fixed sample size this is achieved by propagating each particle from the previous time step just once [Liu98, Gri07a]. Here we propose the following mechanism to generate the particle indexes $i[k]$, for $k = 1, ...M_t$ (recall indexes i indicate which particles from the previous time step $t-1$ are selected to generate the M_t new ones):

$$i[k] : \begin{cases} i[k] = p[k] & k \leq M_{t-1} \\ i[k] \sim \mathcal{U}(1, M_{t-1}) & k > M_{t-1} \end{cases} \tag{8.2.1}$$

Algorithm 2 optimal_pf_selective_resampling $\{x^{t-1,[i]}, \omega^{[i]}\}_{i=1}^{M_{t-1}} \rightarrow \{x^{t,[k]}, \omega^{[k]}\}_{k=1}^{M_t}$

for all particles $x_{t-1}^{[i]}$ **do**
  for $n = 1$ to B **do** // *Generate a set of B samples*
    $x_t^{[n]} \sim p(x_t|x_{t-1}^{[i]}, u_t)$
  end for
  Use the samples to compute $\hat{p}(z_t|\cdot)$ and $\hat{p}_{max}(z_t|\cdot)$
end for
$k \leftarrow 1$
$p \leftarrow \text{perm}([1, 2, ..., M_{t-1}])$ // *Random permutation of the* M_{t-1} *indices*
doResampling $\leftarrow ESS(\{\omega^{[i]}\}_{i=1}^{M_{t-1}}) < 0.5$ // *Selective resampling*
repeat
  if *doResampling* **then**
    Draw an index i with probability given by $\tilde{\omega}_t^{[i]}$. See Eq. (4.3.7)
  else
    if $k \leq M_{t-1}$ **then**
      $i \leftarrow p[k]$ // *Deterministic sampling*
    else
      $i \sim \mathcal{U}(1, M_{t-1})$ // *Uniform (*integer*) sampling*
    end if
  end if
  repeat // *Generate a new sample from i by rejection sampling*
    $x_t^{[k]} \sim p(x_t|x_{t-1}^{[i]}, u_t)$ // *Draw a candidate sample*
    $a \sim \mathcal{U}(0, 1)$ // *Draw a random uniform sample*
  until $a < \Delta$// *Candidate accepted with a probability* Δ *– see Eq. (4.3.10)*
  if *doResampling* **then** // *Now, assign the weight*
    $\omega^{[k]} \leftarrow 1$ // *Reset weights on resampling*
  else
    $\omega^{[k]} \leftarrow \omega^{[k]} \cdot \tilde{\omega}_t^{[i]}$ // *Apply likelihood factor*
  end if
  $k \leftarrow k + 1$
until $k = M_t$// *Number of samples determined by KLD-sampling [Fox03]*

with $p[k] = \text{perm}([1, .., M_{t-1}])$ being a random permutation [1] of the vector $[1, .., M_{t-1}]$, which is computed only once at each particle filter iteration. Therefore, if the number of new particles is smaller than in the previous time step, the permutation assures that each particle has an equal probability of being removed. If the number of particles remains constant, our method becomes the same as in traditional fixed-sized particle filters, disregarding the re-ordering of the samples, which does not affect the estimated density. Finally, in the case of more particles, it is also assured that all the previous particles have the same probability of being selected more than once. Hence, the overall selection method can be seen as sampling from a uniform distribution of indexes, with the advantage of asserting that no hypothesis will be lost as long as $M_t \geq M_{t-1}$ (otherwise it would mean that there are too many particles and we *really want* to remove some hypotheses).

8.3 Evolution of the sample size

As stated above, the adaptive number of samples in our optimal filter is determined by means of the proposal by Fox in [Fox03]. In that method, called KLD-sampling, the continuous d-dimensional state space of the particle filter is approximated by a d-dimensional discrete grid where the number of occupied bins settles the final number of required particles.

In the original context of KLD-sampling, robot localization, the dimensionality of the problem is fixed to $d = 3$ (2-d position plus heading) – which certainly also holds for our first optimal filtering algorithm presented in Chapter 4. Under such a bounded dimensionality we can expect a limited number of samples in most common situations.

[1] The exact algorithm employed for this task is that one implemented in the C++ STL function `random_shuffle`, which assures a uniform distribution over the $N!$ possible permutations of an N-vector (see section 3.4.2 of [Knu81]).

A preeminent (although transient) exception is global localization.

In contrast, in this chapter we face full SLAM, a problem whose dimensionality increases with time t since the filter estimates the whole robot *path*, with a dimensionality of dt, that is, in the order of $O(t)$.

When the number of samples is dynamically adapted in such a problem, it is inevitable for the required number of particles to grow without bounds. This is nothing else than the need of performing importance sampling properly on each of the dimensions, which intuitively can be seen to demand an *exponential* number of samples (this claim is mathematically proven below).

In order to alleviate this rapidly-increasing demand of particles, we propose the following alternative: instead of applying KLD-sampling to the state space of the whole robot path, it can be applied to just the most recent robot pose x_t. Although in this case the dimensionality also stays fixed as in robot localization, there is an important difference: when a robot explores new terrain, its pose uncertainty will grow until a loop closure occurs, hence that in this case the number of samples increases as well.

To sum up, we can devise two ways of computing the adaptive number of samples in our optimal particle filter for SLAM: (i) employ KLD-sampling over the whole robot path, and (ii) use it just over the latest robot pose. It has been shown that, intuitively, both approaches will lead to a growth in the number of samples, although in the first case this growth is unbounded and in the latter the population of particles will reduce after closing a loop.

In order to arrive at a quantitative comparison between both alternatives some simplifications need to be done, given that the exact evolution of uncertainties in SLAM depends on the sensors employed, the properties of the environment and the specific

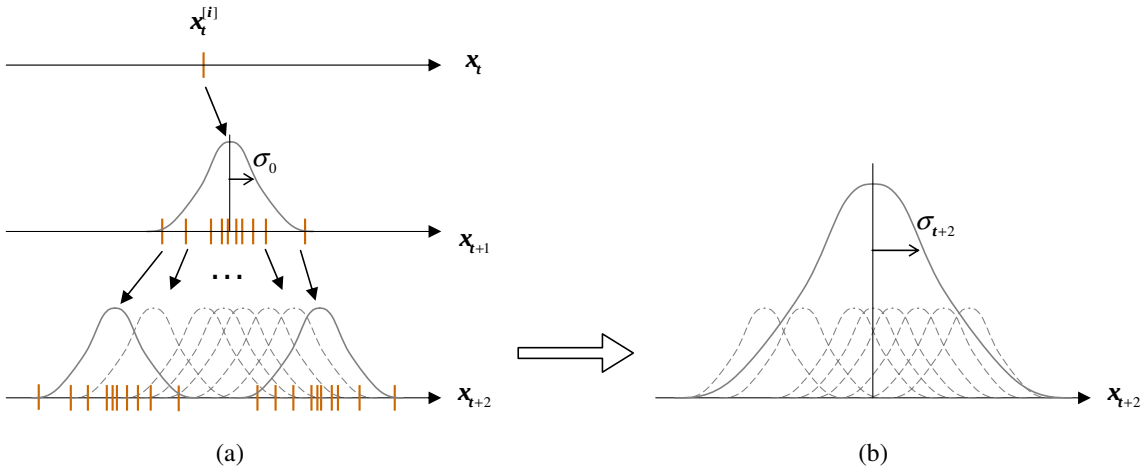


Figure 8.1: (a) The amount of particles required to maintain a proper sampling of the whole state space in full SLAM grows exponentially with time. (b) An alternative proposed in the text, where only the marginal for the latest robot pose is employed to determine the number of required samples.

path followed by the robot. Firstly, we will assume that the estimates for the d dimensions of the robot pose are uncorrelated between them, thus the problem is equivalent to estimating d uni-dimensional variables. Focusing on the case of full SLAM, let's consider the evolution along time of one single particle, as the one represented in Figure 8.1(a) for x_t. After one filter iteration, this single hypothesis should spread over the space due to the combined uncertainties of both transitions and observations. Assuming that this increase of uncertainty "converts" a single particle into a Gaussian with a standard deviation of σ_0, it is obvious that several samples will be required to perform a proper sampling of x_{t+1}. When sampling a Gaussian, it is plausible to assume that KLD-sampling will lead to a number of samples proportional (through a constant k) to its standard deviation.

Therefore, for $t+1$ we have a number of samples $M_{t+1} = k\sigma_0$. Following Figure 8.1(a), it can be seen that for the next time step $t+2$ the process is repeated, but now we have several samples as the start point (those for x_{t+1}). It is easy to see that

$k\sigma_0$ samples will be required to sample x_{t+2} for each "parent" sample in x_{t+1}, from which it can be deduced by induction that:

$$M_t = O\left(k^{dt}\right) \tag{8.3.1}$$

where the factor d needs to be introduced since the above derivation was carried out for just one dimension. This result confirms our intuitive claim at the beginning of this section about the exponential growth in the number of samples required to perform full SLAM.

We now focus on the alternative approximation, where KLD-sampling is applied to just the latest robot pose, x_{t+2} in the present example. Under the same assumptions stated above, it can be shown that the pdf for each uni-dimensional component of the robot pose is a Gaussian, whose variance can be computed by means of a Kalman filter, and being $\sigma_t \propto \sigma_0\sqrt{t}$. As above, if KLD-sampling gives us a number of required particles linear with the standard deviation of the Gaussian being sampled, we arrive at $M_t \propto \sqrt{t}$, or, considering the d dimensions of each robot pose:

$$M_t = O\left(t^{\frac{d}{2}}\right) \tag{8.3.2}$$

Given the drastic reduction in the evolution of the number of samples, from being exponential to being polynomial, we have considered employing this second alternative in our experiments.

An important final remark is that these growths in complexity are not a characteristic of our proposed method, but of any RBPF-based approach to SLAM in general. In fact, the analysis presented in this section has made patent the degree of sub-optimality of *all* previous works where the number of particles always remain constant.

8.4 Experimental evaluation and discussion

This section demonstrates an application of the proposed algorithm to occupancy grid mapping through a RBPF. For comparison, our approach is contrasted to a recent proposal [Gri07a] which approximates the observation model by a Gaussian through scan matching (SM) – we will refer to this approach as SM-based particle filter. The non-parametric observation model employed here is the likelihood field, described in [Thr05].

The sensory data used as the input to the filters are a part of the Málaga University campus dataset [2] , where our mobile robot SENA [Gon06b] closes a loop of about 60 meters in a semi-outdoor scenario, moving around one building. Snapshots of the evolution of the PF using our algorithm can be seen in Figure 8.2(a)–(d) at different time steps. The final map obtained for the SM-based filter, which used a fixed sample size of 15, is represented in Figure 8.2(e). All those maps are the most likely grids at each time step, overlapped with all the robot path hypotheses in the filter.

Note that the loop closure, which happens approximately at time steps 140-160, reduces the uncertainty in the robot path in both methods, though it is managed differently in our method and in the SM-based filter. In the latter, since the number of particles remains constant while estimating a space with a growing dimensionality (the robot path), the samples will become more and more sparse with time, providing a poorer representation of the actual state space. As a result, after closing the loop just a few hypotheses will survive. The problem that arises with this approach is that typically only one hypothesis remains for a large part of the robot path: its uncertainty has been lost and there is not a well-defined probability distribution for the path. This can be observed for the concrete case of this experiment, in Figure 8.2(e). In contrast,

[2] Available online: http://babel.isa.uma.es/mrpt/downloads/.

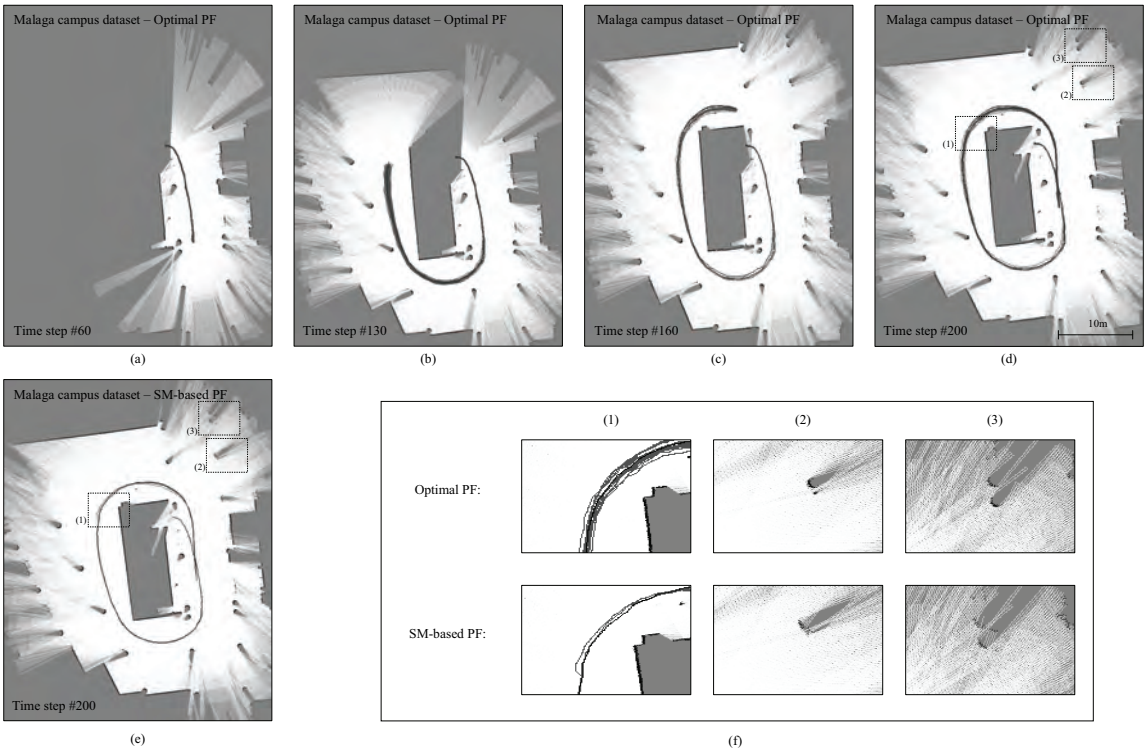


Figure 8.2: Experimental results for map building. (a)–(d) Four snapshots of the evolution of our optimal filter while building the map for the Málaga campus dataset. (e) The final result for a SM-based particle filter, for comparison. (f) Detailed views of certain areas of the final paths and maps for our optimal algorithm and the SM-based PF. Inconsistencies can be observed in the latter.

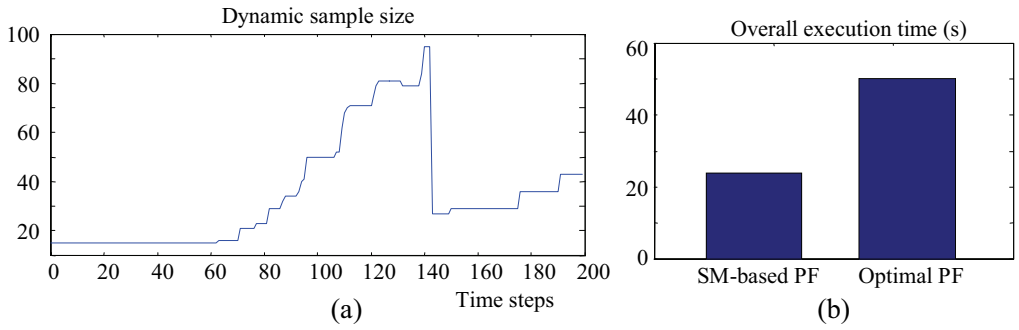


Figure 8.3: (a) The number of particles dynamically chosen by our method. The sample size increases until the loop is closed, about time step #140. (b) A comparison of the overall computation times taken by both methods.

our filter dynamically increases the number of samples as the robot moves along the loop in order to provide a proper sampling of the state space. Observe how this number decreases after closing the loop, a reflection of our usage of the alternative approximation described in §8.3 to estimate the number of required samples at each time step. Notice as well that there are more path hypotheses before closing the loop than after it (refer to Figure 8.2(b)–(c), respectively). The complete evolution of the sample size is sketched in Figure 8.3(a).

The memory usage of the filter closely follows the same evolution, since each particle carries a hypothesis for the whole map: in our method both the computation and memory requirements increases as the robot follows longer loops.

Additionally, the SM-based filter may lead to inaccuracies due to small errors in the scan matching procedure (refer to the discussion in section 4.1). We have highlighted some areas in the resulting maps to illustrate that our optimal filter gives a more precise robot localization, which turns into maps without the inconsistencies that do appear for the SM-based filter in both the path and the map. Refer to the detailed views in Figure 8.2(f).

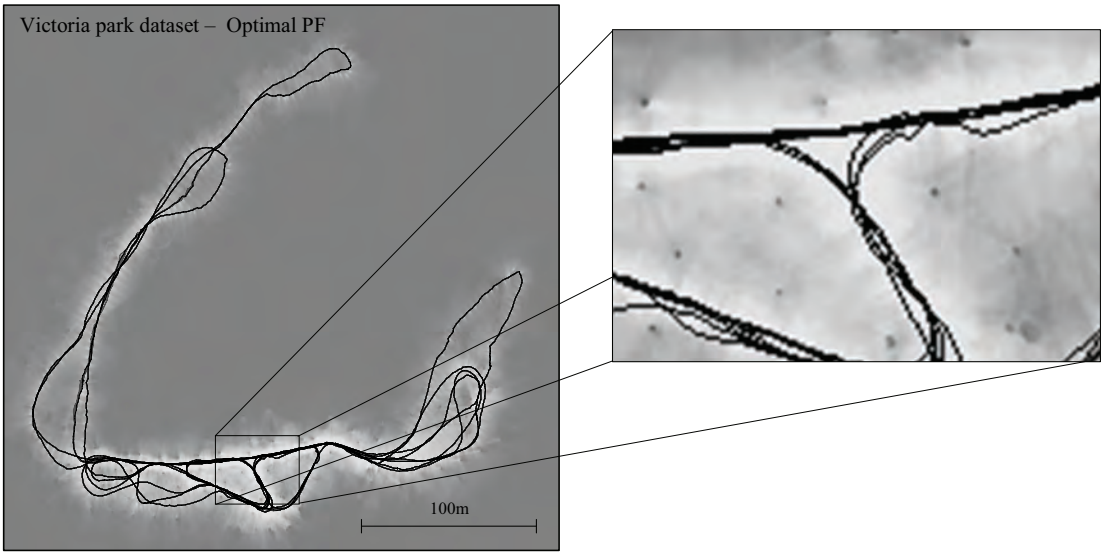


Figure 8.4: The map built for the Victoria Park dataset using our optimal filter.

Regarding the computation time of our method, it takes 50.2s to process the whole dataset, while the SM-based approach only needs 23.8s – see Figure 8.3(b). Thus, the accuracy of our optimal filter is gained at the cost of both a higher computational complexity and memory consumption. Therefore, if a given application requires a very fast processing time, SM-based methods such as [Gri07a], or the even more optimized version in [Gri07b], should be used. If it is more important to assure the consistency of the estimation and to obtain more accurate maps, then our optimal filter should be preferred.

The method has also been tested with an outdoor scenario, in particular, using the standard dataset gathered at Sydney's Victoria Park [Gui01]. The most likely grid map at the end is shown Figure 8.4, along with a detailed view of the central part of the environment, where the trees can be appreciated clearly. To the best of our knowledge, this is the first time a grid map has been built for this dataset, which has been largely used to test EKF-like SLAM methods. In spite of a greater memory usage than in a

landmark map, using a grid map avoids the problems of landmark (tree) detection and data association, and exploits the most of the information in the laser scans whereas in a EKF-like approach only the trees, once detected, could be used to localize the robot.

CHAPTER 9

AN EFFICIENT SOLUTION TO RANGE-ONLY SLAM

9.1 Introduction

Most works in the field of SLAM focus on sensors that provide either range and bearing information (e.g. the "classic" solution to SLAM [Dis01]) or bearing-only information (e.g. Mono-SLAM [Civ08, Dav07]). Relatively few works have addressed the problem of building maps with range-only (RO) sensors in despite of the important applications where they are an appropriate choice, such as submarine autonomous vehicles [New03] or ground vehicles in industrial environments [FM07]. There are two fundamental characteristics that render RO-SLAM specially challenging: the existence of outliers due to the sensor nature (typically sonar or radio pulses), and more importantly, the high ambiguity of the measurements (in the sense that we have no bearing information

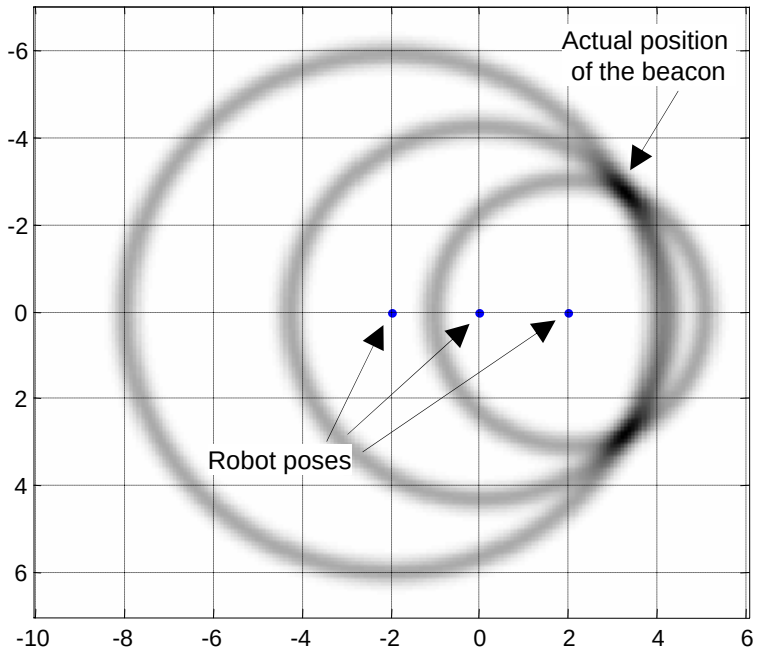


Figure 9.1: One peculiarity of range-only SLAM is that map estimations may converge to multi-modal densities. In this example, the symmetry in the distance (range) observations made by a robot to one fixed beacon as it travels over a straight path eventually leads to two regions with a high probability of containing the sensed beacon.

at all). This later issue is illustrated in Figure 9.1, where a robot measures its distance to one fixed beacon from three different positions along a straight path. For each position it is shown the ring-shape area of the estimated position of the beacon that is consistent with the measurement. Concretely, the figure represents the probabilistic *inverse sensor model* (refer to Chapter 7).

Therefore, these sensors carry two hurdles: (i) the large portion of the environment where a beacon could be, given just one observation, and (ii) a very likely possibility of multiple plausible hypotheses. These issues become more severe when building 3-d maps, where beacons can be placed at different heights. Thus, RO-SLAM may become a problem even more challenging than bearing-only SLAM ([Dav07,Kwo04]), where the landmark estimates typically converge to a single mode. On the other hand, depending on the sensor technology, RO-SLAM has the advantage of avoiding the data association

problem, since beacons usually can self-identify themselves.

Regarding previous proposals for RO-SLAM, in [Sin02] it is reported a geometric method for adding new beacons to a map using delayed initialization, but a partially known map is required at the beginning. Sub-sea RO-SLAM is demonstrated in [New03] with good results, even with the lack of a reliable ego-motion estimation (e.g. from odometry). The main difference with the present work is the usage of a least-square error minimization procedure instead of a probabilistic filter where several robot path hypotheses can be considered simultaneously. The work in [Ols04] achieves RO-SLAM through a different strategy: firstly, an initial estimation of the position of each beacon is computed using a voting scheme over a 2D grid, then a standard EKF deals with the SLAM problem. A similar scheme is adopted in [Dju06], where the authors also explore the possibility of inter-beacon range measurements to improve map building. A recent work [Dju08] is similar to ours but formulated in polar coordinates, hence requiring a reduced number of Gaussians. However, that approach raises the issues of when to create new hypotheses from noisy range intersections, and whether the linear approximation of uncertainty in an EKF suffices to capture the actual large uncertainty of predicted ranges, an interesting point worthy of further analysis.

We reported a probabilistic formulation of RO-SLAM based on a Rao-Blackwellized particle filter (RBPF) in [Bla08e]. This approach, also adopted in the method discussed in this chapter, has the advantage of decoupling the conditional distributions of each beacon in the map for each path particle, which entails freedom in the design of each of these distributions in such a way that at some given instant several already well-localized beacons may coexist in the map with more recently added beacons that have higher uncertainty.

The contributions of the new method described here are: (i) a new inverse sensor model for initializing map distributions as weighted Sums of Gaussians (SOGs), (ii)

the explanation of how to update those Gaussians and their weights using a multi-hypothesis EKF, and (iii) the derivation of the corresponding observation model required for the RBPF.

The present approach has the advantage of always providing an immediate estimate without delayed decisions, while still providing an accurate approximation of the strongly non-Gaussian and frequently multi-modal distributions found in RO-SLAM: all the available information is exploited to perform localization without waiting until the beacon distributions converge as in other proposals. In addition, new beacons can be inserted at any time in the filter, which is unfeasible under EKF-based solutions to RO-SLAM that need a first stage until the distributions can be properly approximated by Gaussians. Finally, a statistical analysis from simulations is also provided demonstrating that the presented approach can cope with highly noisy sensors and reduces in one order of magnitude the average errors of our previous method proposed in [Bla08e] at a fraction of its computation time.

The rest of this chapter is outlined as follows: the next section provides the notation and background required to our formulation of RO-SLAM, which is discussed in depth in §9.3 and subsections therein. The final section presents diverse experiments with both synthetic and real data in order to validate our method.

9.2 Problem statement

Firstly, we briefly review the notation of the variables involved in the problem. We denote the robot path as the sequence of poses in time $x^t = \{x_1, ..., x_t\}$, while robot actions (odometry) and observations (range measurements) for each time step t are represented by u_t and z_t, respectively.

As already mentioned in Chapter 7, the RBPF approach to estimating the SLAM

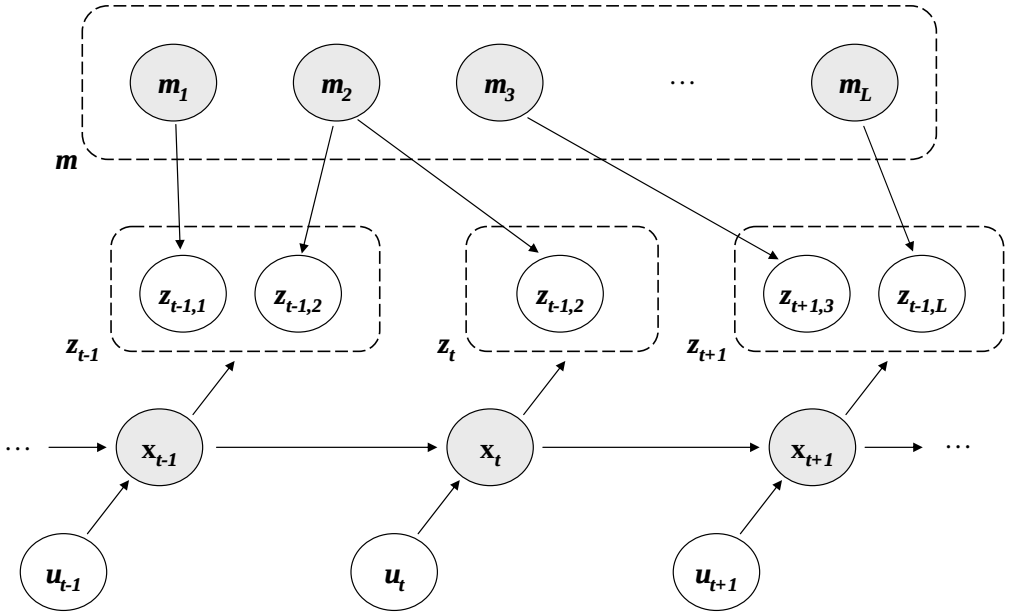


Figure 9.2: The dynamic Bayesian network for mobile robot RO-SLAM, where robot poses x_t and the map m are hidden variables (represented as shaded nodes) to be estimated from actions u_t and sensor observations z_t. The map comprises a set of variables $\{m_1, m_L\}$, one for each individual beacon.

joint posterior $p(x^t, m|u^t, z^t)$ consists in approximating the marginal distribution of the robot path x^t using importance sampling, then computing the map as a set of conditional distributions given each path hypothesis. We denote the i'th particle as $x^{t,[i]}$ and its corresponding importance weight as $\omega_t^{[i]}$. By choosing a RBPF approach to RO-SLAM, we can factor the distribution of the map associated to each particle as:

$$p\left(m|x^t, z^t, u^t\right) = \prod_{l=1}^{L} p\left(m_l|x^t, z_l^t\right) \tag{9.2.1}$$

m_l being the different individual beacon positions in the map m. The factorization in Eq. (9.2.1) follows from the conditional independence of each beacon in the map given the robot path and the observations. From the DBN of the problem, in Figure 9.2, it can be inferred that $\{x^t, z^t\}$ d-separates (see §2.10) any pair of beacons in the map, that is,

$$m_i \perp\!\!\!\perp m_j \mid x^t, z^t \qquad \forall i \neq j \tag{9.2.2}$$

As a consequence of this conditional independence between individual beacons in the map, we can employ the most convenient representation for their pdfs at each time step without affecting either the robot path or the other beacons. Section 9.3 is devoted to the computation and the update of these densities within the main RBPF.

For completeness, the sequential algorithm to be executed at each time step is summarized next. Assume that the set of particles for the previous time step $x^{t-1,[i]}$ are approximately distributed according to the real posterior. In the case of the first time step, all the particles can be arbitrarily initialized to the origin. In subsequent iterations, new particles are drawn using the robot motion model (in our case, derived from odometry readings), that is, $x_t^{[i]} \sim p(x_t|x_{t-1}^{[i]}, u_t)$. Next, the importance weights are updated as $\omega_t^{[i]} \propto \omega_{t-1}^{[i]} p(z_t|x^{t,[i]}, z^{t-1})$ with the probabilistic observation model $p\left(z_t|x^{t,[i]}, z^{t-1}\right)$ that will be derived in §9.3.4. If necessary, the particles may be resampled to preserve their diversity. This is typically performed whenever the effective sample size falls below a given threshold [Liu96]. After updating the estimate of the robot path, the corresponding conditional distributions of the map must be also updated to account for the new range readings, as discussed in the next section.

Notice that the algorithm described above corresponds to a standard SIR particle filter (see §4.1), although the proposed approach to RO-SLAM is also wholly compatible with the optimal filtering algorithm presented in Chapter 4.

9.3 Proposed solution

As already mentioned, each robot path hypothesis is associated with the corresponding conditional map distribution $p(m_l|x^{t,[i]}, z^t)$ for each beacon l. Note that in the following we drop the l subscript for clarity, since subsequent derivations apply equally and independently to any number of beacons in the map.

Incorporating new information (the new robot pose x_t and the observation z_t) into the map belief m of one beacon is carried out by applying the Bayes rule, as follows:

$$\begin{aligned} \underbrace{p(m|x^t, z^t)}_{\text{Posterior}} &\overset{\text{Bayes rule}}{\propto} p(m|x^{t-1}, z^{t-1}) p(x_t, z_t|m, x^{t-1}, z^{t-1}) \\ &= p(m|x^{t-1}, z^{t-1}) \underbrace{p(x_t|m, x^{t-1}, z^{t-1})}_{\text{Constant among all particles}} p(z_t|m, x^t, z^{t-1}) \\ &\propto \underbrace{p(m|x^{t-1}, z^{t-1})}_{\text{Bayes prior}} \underbrace{p(z_t|m, x^t, z^{t-1})}_{\text{Inverse sensor model}} \end{aligned} \tag{9.3.1}$$

where the marked term is constant among all the RBPF particles since neither the last action u_t nor the last observation z_t appear. Put in words, the posterior belief is obtained by multiplying the previous belief by the inverse sensor model, which determines the likelihood of finding the beacon at any position m given the observed range value z_t and assuming x_t as some fixed hypothesis for the robot pose.

The next subsections address the maintenance of the beacon pdfs within the map and how to employ them to derive the sensor observation likelihood function.

9.3.1 Inserting a new beacon

When a beacon is firstly observed at some given instant of time t (not necessarily the first time step), the prior belief in Eq. (9.3.1) is undefined. If there is no a priori information about the spatial disposition of beacons it is plausible to assume a

uniform prior. There is however one interesting exception with important practical consequences: in the case of a ground vehicle that is building a 3-d map, if we know in advance that all the beacons have been placed at a height above (or below) the robot (a typical situation for an industrial environment), the prior becomes a uniform distribution over half of the space and zero in the complementary part. This is important since, as will be illustrated with experiments, a vehicle moving on a flat, horizontal scenario can build a 3-d map only up to a symmetry with respect to the robot plane: it cannot be disambiguated whether a given beacon is above or below the robot.

Once the prior belief has been defined as a uniform pdf, it follows from Eq. (9.3.1) that the initial map distribution becomes simply the inverse sensor model $p(z_t|m, x^t, z^{t-1})$ (or its evaluation over half of the 3-d space in the special case commented above). Assuming a range sensor model with additive Gaussian noise of variance σ_s^2, the inverse observation model for a beacon m is:

$$p(z_t|m, x^t, z^{t-1}) \propto \exp\left(-\frac{1}{2}\frac{|x_t - m|^2}{\sigma_s^2}\right) \tag{9.3.2}$$

The evaluation of this model gives the typical fuzzy-ring shapes associated to RO-SLAM. The problem is that this density cannot be filtered iteratively in an analytical form if the beacon locations are represented in Cartesian coordinates [1], hence we must rely on approximations. In this work we propose to adopt a Sum of Gaussians (SOG) approximation[2] to the exact ring-like shape, such as:

$$p(z_t|m, x^t, z^{t-1}) \approx \sum_{k=1}^{N} \upsilon_t^k \mathcal{N}(m; \hat{m}_t^k, \Sigma_t^k) \tag{9.3.3}$$

[1] Refer to the comments in §9.1 about the alternative polar representation of [Dju08] and its drawbacks.

[2] The effects of this approximation are quantified below in this section – refer to the discussion on the Figure 9.4.

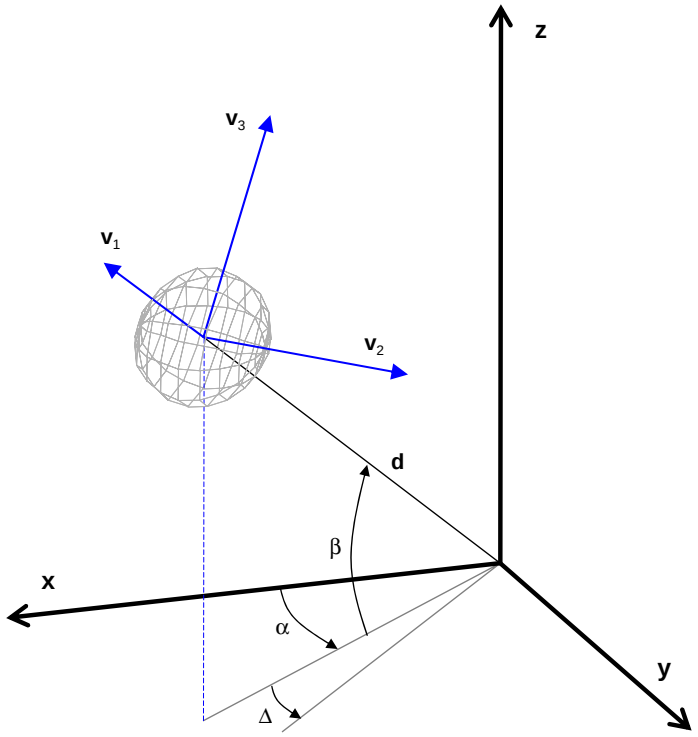


Figure 9.3: The variables involved in the generation of the Gaussian modes within each SOG of the inverse sensor model. Azimuth and elevation angles from the sensor position (the coordinate origin in the figure) are represented by α and β, respectively. The covariance matrix is computed by mapping the uncertainties in the radial ($\mathbf{v}_1$) and tangent ($\mathbf{v}_2$, $\mathbf{v}_3$) directions using the appropriate transformation matrix. Refer to section 9.3.1 for further details.

being υ_t^k the weight of each individual Gaussian, and $\hat{m}_t^k$ and Σ_t^k its mean and covariance matrix, respectively.

In particular, for a given range measurement $z_t = r$, we have to generate a number of Gaussians equally spaced over a sphere centered at the sensor position and with radius r (see Figure 9.3), following the procedure described next[3] Each of the individual Gaussians is thus placed at a direction stated by a discrete set of azimuth ($-\pi < \alpha \leq \pi$) and elevation ($-\pi/2 < \beta \leq \pi/2$) angles. Let Δ denote the angular increments between consecutive Gaussians along either α or β. Since the model approximation will become

[3]An alternative approach would be to consider a distribution of Gaussians following a spiral as described in [Saf97].

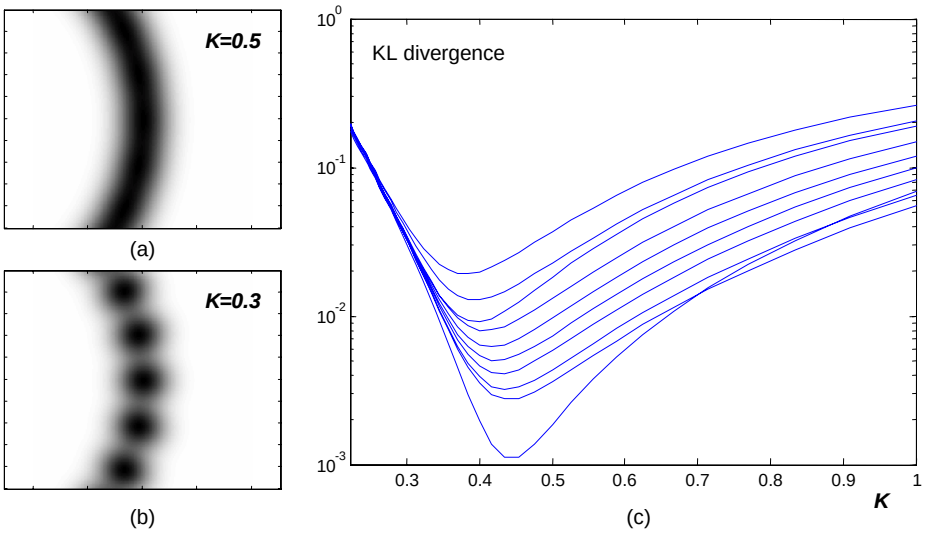


Figure 9.4: (a)–(b) Inverse sensor model for a sensed range of $r = 1$ meter computed from our SOG approximation (simplified here to 2-d for clarity) and two different values of the constant K, which determines the standard deviation of Gaussian nodes in tangential directions. (c) The Kullback-Leibler divergence (KLD) of our SOG approximation as a function of K and for r between 1 and 10.

poorer for larger distances between Gaussians, let d_m represent the maximum distance allowed between adjacent Gaussians in the regular grid over the sphere. Under this constraint, the angular increment Δ can be computed as $\Delta = 2\pi/B$, being B the integer number of modes along any great circle of the sphere, determined as $B = 2\lceil \pi r/d_m \rceil$, where the factor 2 is out of the ceiling operator to force an even number of modes and therefore to assure symmetry.

At this point we have computed the discrete sequences of angles α_i and β_j for $i = 1, .., B$ and $j = 1.., \frac{B}{2}$ that define a regular grid over a sphere centered at the sensor. Thus, the mean of each SOG mode is given simply by:

$$\hat{m}_t^{ij} = \begin{pmatrix} x_0 + r\cos\alpha_i \cos\beta_j \\ y_0 + r\sin\alpha_i \cos\beta_j \\ z_0 + r\sin\beta_j \end{pmatrix} \tag{9.3.4}$$

Here $(x_0\ y_0\ z_0)^T$ stands for the absolute coordinates of the robot range sensor, which

are known since we are assuming a robot pose hypothesis $x_t^{[i]}$.

To compute the covariance Σ_t^{ij}, let's define three unit orthogonal vectors with origin the mean of the Gaussian. For convenience, the first vector $\mathbf{v}_1$ will be always pointing radially, while the others ($\mathbf{v}_2$ and $\mathbf{v}_3$) are tangential to the sphere, as illustrated in Figure 9.3. Such a vectorial base is well-defined for any sphere of non-zero radius. Note that the uncertainty in the radial direction (the "thickness" of the sphere) is determined by the noise σ_s in the sensor model stated by Eq. (9.3.2). Since we desire radial symmetry by design, the uncertainty σ_t in both tangential directions should be equal and proportional to the separation between Gaussians, that is, $\sigma_t = rK\Delta$, with K being a proportionality factor. The covariance is then built as:

$$\Sigma_t^{ij} = \begin{pmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \mathbf{v}_3 \end{pmatrix} \begin{pmatrix} \sigma_s^2 & 0 & 0 \\ 0 & \sigma_t^2 & 0 \\ 0 & 0 & \sigma_t^2 \end{pmatrix} \begin{pmatrix} \mathbf{v}_1^T \\ \mathbf{v}_2^T \\ \mathbf{v}_3^T \end{pmatrix} \tag{9.3.5}$$

Finally, the weights υ_t^{ij} of all the generated Gaussians in Eq. (9.3.3) are equal since all of them are equally probable. It is worth noting that the above process can be also applied to 2-d maps by fixing β to 0 and discarding one tangent vector.

The constant K deserves further comment since it determines the quality of the approximation to the real inverse sensor model. To illustrate graphically the effects of this constant, please refer to Figure 9.4(a)–(b), where a SOG is generated for $K = 0.5$ and $K = 0.3$ and a 2-d map model. Simple visual inspection reveals that the second case leads to a worse approximation of the actual "ring" density. We have computed the Kullback-Leibler divergence (KLD) (see §2.6) as a proper measure of similarity between the approximate and the actual densities for different measured ranges r from 1 to 10 meters, and for different values of K. The results, in Figure 9.4(c), reveal that good approximations can be obtained, but unfortunately the optimal K varies with

the sensed range r. However, values of K near 0.4 lead to good approximations, hence it is the one employed in our experiments.

9.3.2 Updating the map SOG distribution

Once a beacon has been inserted in the map, subsequent range measurements z_t update its belief through Eq. (9.3.1), which in this particular case of the prior density defined as a SOG can be efficiently implemented as a multi-hypothesis EKF. Basically, the mean and covariance are updated using a standard EKF [Jul97], while the weights are updated by evaluating the actual reading z_t into the Gaussian of the observation predicted by each SOG mode [Als72], that is,

$$\upsilon_t^k \propto \upsilon_{t-1}^k \mathcal{N}(z_t; h_t^k, {\sigma_t^k}^2) \tag{9.3.6}$$

where the mean $h_t^k = h(x_t^{[i]}, \hat{m}_t^k)$ is computed using the sensor model $h(\cdot)$, and the variance ${\sigma_t^k}^2$ includes the sensor noise σ_s and the projection of the beacon uncertainty:

$${\sigma_t^k}^2 = H \Sigma_t^k H^T + \sigma_s^2 \tag{9.3.7}$$

being H the Jacobian of the sensor model. An important insight into this approach is that, eventually, most of the Gaussians in a SOG will have negligible weights as they become inconsistent with the complete history of observations. Since the contribution of these modes to the overall density will be negligible as well, we can simply remove them from the SOG when their weights fall below a given threshold. Thus, our approach can automatically adapt its computational burden to the actual uncertainty present at each instant of time, as will be shown later on with real experiments (this is analogous to approaches that scale the number of samples in particle filters [Fox03]).

Another alternative to simply deleting Gaussian modes is to apply SOG reduction

methods such as that proposed in [Run07], where two SOG modes are "fused" only if the lost information is below a certain threshold. This approach has not been implemented, and in spite of possibly yielding good results, it would also carry a higher computational load than the simpler method of removing very unlikely modes.

9.3.3 An illustrative example

To illustrate and clarify all the ideas discussed up to this point, consider the example depicted in Figure 9.5, where a perfectly localized robot estimates the position of just one beacon (i.e. there is only mapping, no localization). At the initial instant t_1, the SOG is created as the robot observes the beacon for the first time, following the procedure described in §9.3.1 (in this case reduced to 2D for clarity in the representation). Next, as the robot moves along a straight path, the mean and covariances of all the Gaussians are modified by subsequent observations, as can be appreciated at instant t_2. At this point many modes already have negligible weights, hence they have been removed from the filter. Later on, at time t_3, the beacon has converged to two symmetrical modes with respect to the robot path. The symmetry is finally broken when the robot moves away from its previous straight path, as it can be seen in t_4.

9.3.4 The observation likelihood model

Taking into account our approximation of each map distribution as a SOG in Eq. (9.3.3), we can expand the observation model required to update the weights of the RBPF as follows:

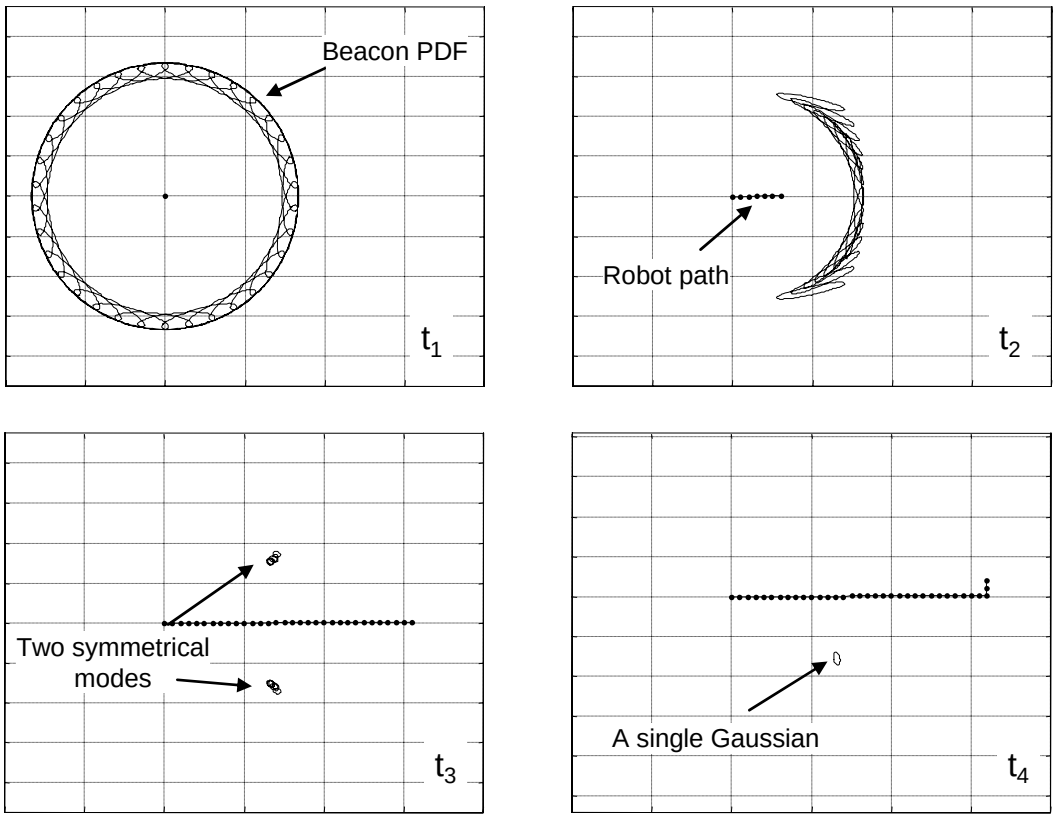


Figure 9.5: An example of how our approach iteratively estimates the location of one beacon. It is shown how the Gaussian modes with lowest weight in the SOG are discarded over time, and how symmetrical results are obtained for straight paths. In t_4 it can be seen how this symmetry quickly disappears when the robot deviates from the straight path.

$$\begin{aligned} p(z_t|x^{t,[i]}, z^{t-1}) &= \int_{-\infty}^{\infty} p(z_t|x_t^{[i]}, m) p(m|x^{t-1,[i]}, z^{t-1}) dm \\ &= \sum_{k=1}^{N} v_{t-1}^k \mathcal{N}\left(z_t; h_t^k, \sigma_t^{k\,2}\right) \end{aligned} \tag{9.3.8}$$

where each normal distribution represents the predicted observation for one mode within the SOG. The parameters of these Gaussians have been already described in Eq. (9.3.6)–(9.3.7).

As an implementation note, it has been already mentioned (§4.3.2) the convenience of considering *logarithmic* weights and likelihood. In this case, implementing Eq. (9.3.8)

requires recovering the non-logarithmic weights and thus it is prone to numeric overflow. A numerically-stable method is described in Appendix B.2 that avoids this issue, thus greatly extending the dynamic range of all the involved values.

9.4 Experimental evaluation and discussion

In order to validate the proposed approach, we have performed extensive simulations in order to (i) characterize its performance against different levels of sensor noise, and (ii) to compare it with one of our previous work on RO-SLAM [Bla08e] which did not rely on a SOG representation for the beacons. Additionally, we present the construction of a 3-d map from data gathered by a real robot. A video illustrating the following results and the source code are available online in [Bla10a].

9.4.1 Performance characterization

Firstly, we have carried out three series of simulations in order to characterize statistically the accuracy of the maps generated by the present approach. The first experiment characterizes the average localization error for beacons in the final map as a function of the sensor noise σ_s. The results are represented in Figure 9.6(a) through the mean errors and their corresponding 67% confidence interval. To obtain statistically significant results we have executed our approach 50 times for each parameter value, with 20 randomly placed beacons each time. Average errors as a function of odometry noise and the number of particles in the RBPF have been computed similarly, and the results are plotted in Figure 9.6(b)–(c). Errors in odometry are represented as the ratio between the standard deviation of the Gaussian noise and the actual increment between consecutive robot poses.

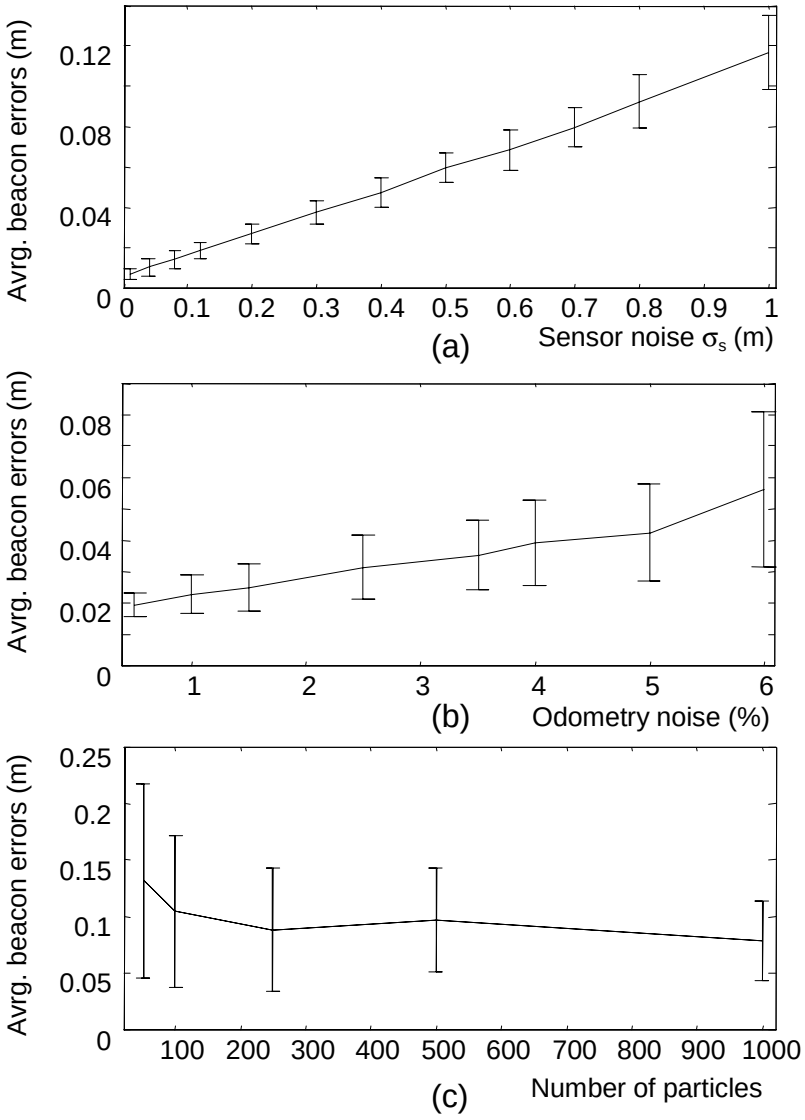


Figure 9.6: Statistical results from simulations sweeping different parameters: (a) the sensor noise (σ_s), (b) odometry errors, and (c) the number of particles in the RBPF. Results present average errors for beacon locations among 67% confidence intervals.

The interpretation of these statistical results is that, as expected, lower levels of noise or more particles lead to lower average errors. Our method can cope with heavily corrupted range observations (e.g. σ_s up to $1m$ in a $20 \times 20m$ map) exhibiting map errors approximately proportional to the sensor noise (see Figure 9.6(a)). However, quite small errors in odometry – above 5% in our experimental setup, see Figure 9.6(b) – lead to a faster raise in the map errors. The reason is the usage of the standard proposal in the RBPF, while the more optimal choice presented in Chapter 4 would give increased robustness at the cost of a larger computational burden.

9.4.2 Comparison to a the Monte-Carlo approximation

We have compared our method to the previous work [Bla08e], where a Monte-Carlo (MC) approximation was employed instead of the SOG. In order to provide a fair comparison between both methods, we have analyzed the average errors in the maps for similar computational burdens and the average execution times for similar map errors. The results are summarized in Figure 9.7(a) and (b), respectively. For similar computation times, the MC approach obtained an average error of 0.28m while for our new proposal this error reduces to 0.03m. Furthermore, in the second situation (similar map errors) the average execution times are 3.9 and 32.4 ms per particle for the SOG and MC solutions, respectively. Therefore, the new proposal outperforms [Bla08e] by one order of magnitude in both efficiency and accuracy.

We also have analyzed the tolerance of both methods to outliers in the range data, which in our approach are specially problematic if we get an outlier the first time a beacon is observed, since we rely on this first range to initialize the map distribution (see §9.3.1). With this purpose, we have set up an experiment simulating a highly noisy sensor, $\sigma_s = 1m$, whose ratio of outliers is 30% the first time a beacon is observed and

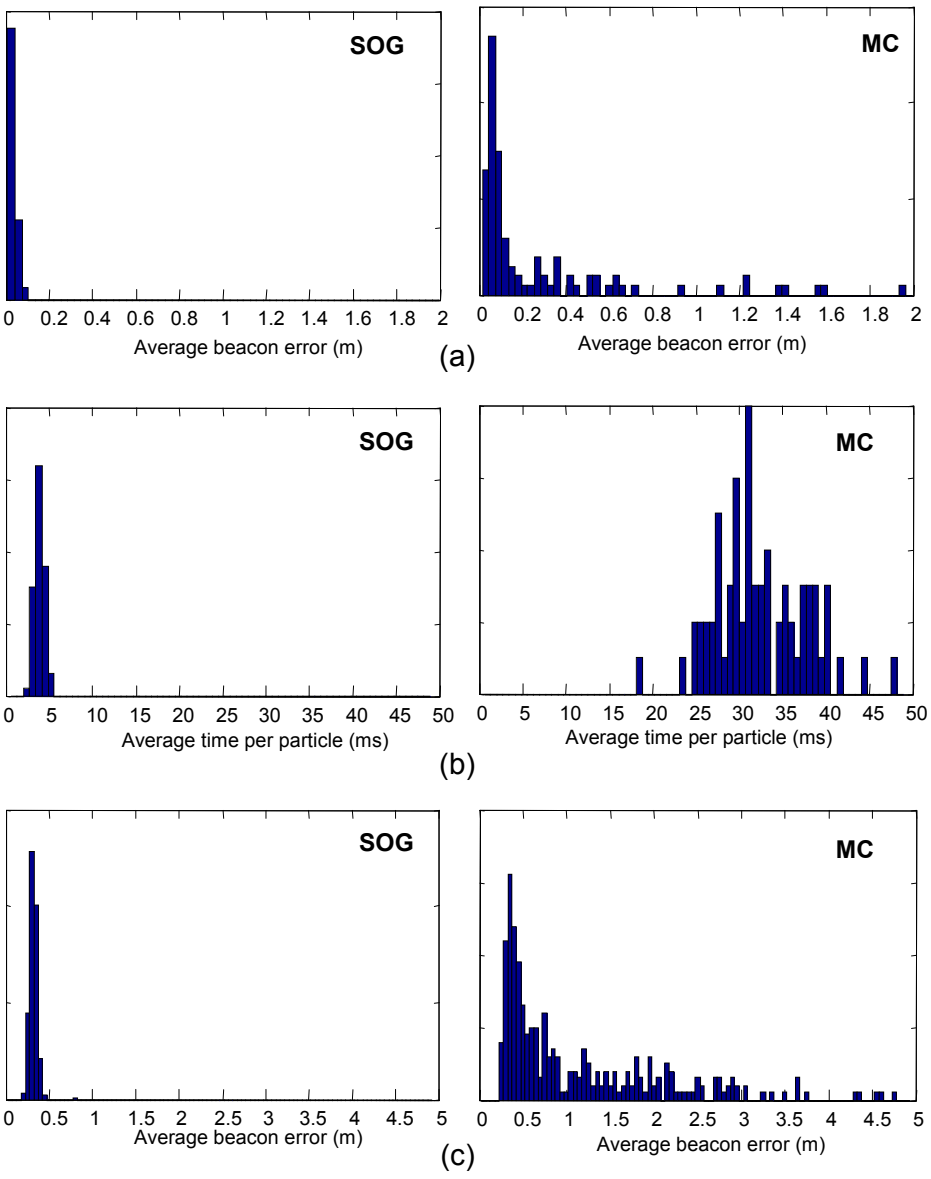


Figure 9.7: A comparison of the new presented approach (SOG) and the method in [Bla08e] (MC). (a) Average beacon error (m) for similar computation time in both approaches. (b) Average computation time per particle (ms) for similar final beacon errors. (c) Beacon errors for a simulated sensor heavily corrupted with noise and outliers.

5% otherwise. Outliers have been simulated by adding to the range measurements a uniform noise between 1 and 10 meters. The average map error histograms for 50 repetitions are shown in Figure 9.7(c). In this case, a mean error of 1.12m is obtained for the MC approximation, while the present approach achieves a significant reduction to 0.32m. These better results of the SOG representation arise from the better adaptability of Gaussian modes to newer observations, i.e. they can be "displaced", even recovering from a heavily corrupted initial distribution.

9.4.3 Evaluation with a 3-d map from a real dataset

Finally, we have applied our approach to a sequence of UWB range measurements [FM07] within an indoor environment. The experimental setup consists of a Pioneer mobile robot with a UWB transceiver onboard, while other three UWB devices act as static radio beacons. The position of the three beacons has been measured manually to provide the ground truth required to evaluate the results. In this case we have employed the a priori knowledge that beacons are above the robot in order to limit the map prior distribution to one half of the 3-d space, as discussed in section 9.3.1.

After completing one loop along the room, the estimates for all three beacons have converged to unimodal distributions, as shown in Figure 9.8(b)–(c). In despite of the noisy sensor ($\sigma_s = 0.10m$), it can be observed in Figure 9.8(a) how the errors in the location of each beacon quickly vanish as the robot moves just a few meters. The better estimation of the z-coordinate in the case of beacon #3 (refer to Figure 9.8(c)) is a consequence of its higher height relative to the robot, which makes the range observations less ambiguous. The errors between the final beacon estimates (their means) and the ground truth are summarized in Table 9.1.

Finally, observe in Figure 9.9(a)–(b) how the computational burden of the algorithm

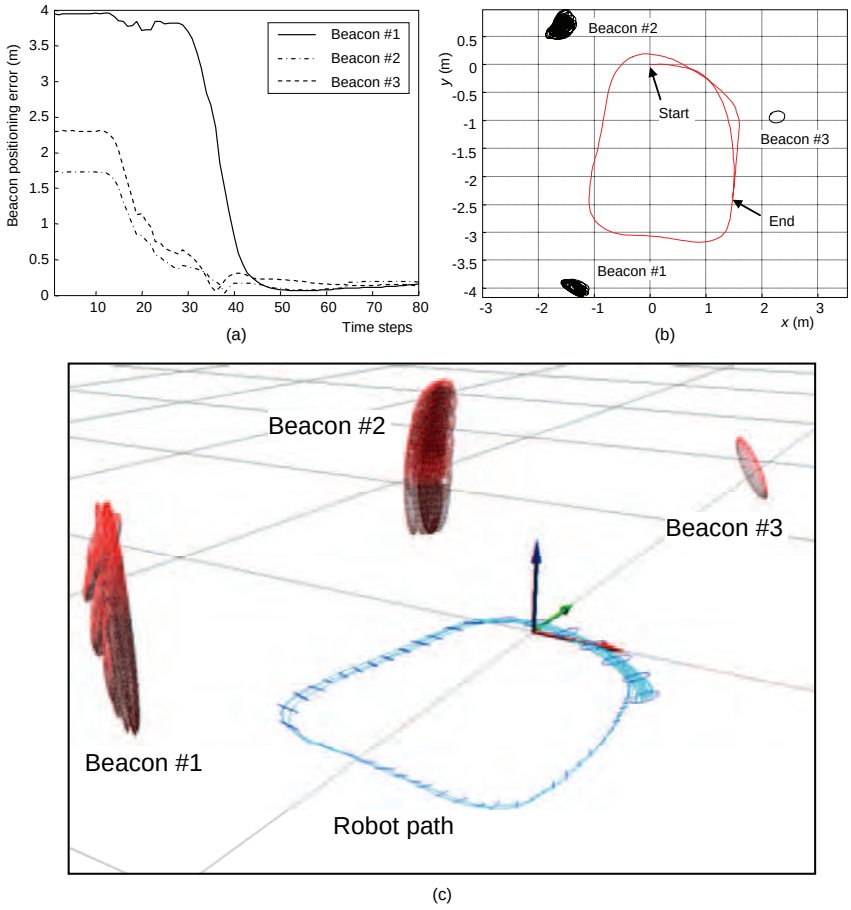


Figure 9.8: Results from data gathered by a real robot equipped with a UWB transceiver. (a) Errors in each beacon estimated localization with respect to the ground truth. (b)–(c) A bird-view and a 3-d representation of the final state of the filter, respectively. The map shown is the one associated to the particle with the highest probability.

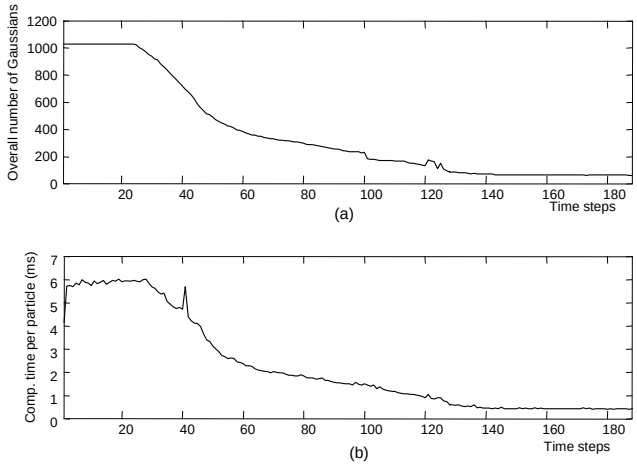


Figure 9.9: Results from data gathered by a real robot equipped with a UWB transceiver. (a)–(b) Overall number of Gaussians in the most likely map and computation time, respectively, for each iteration of the algorithm.

Table 9.1: Summary of Errors for the 3-d Map Built from UWB Data

Beacon coordinate		Ground truth (m)	Estimate (m)	Error (m)
#1	x	0	-0.059	0.059
	y	0	-0.278	0.278
	z	0.912	1.17	0.257
#2	x	-0.320	-0.392	0.072
	y	4.332	4.419	0.087
	z	1.374	1.214	0.159
#3	x	3.403	3.513	0.109
	y	2.802	2.740	0.062
	z	2.175	2.108	0.067

decreases with time as the map estimate becomes more precise and fewer Gaussians are required to represent the maps densities. Unlike in our previous MC-based approach [Bla08e], a reduced number of Gaussians will not lead to the degeneracy of the filter. At the point of maximum complexity, just after initializing the three beacons, the computation time is 6ms per particle. Thus, our method is efficient enough for performing online on a real robot.

9.4.4 Discussion

In this chapter we have presented a novel representation for the map conditional densities associated to the particles of a RBPF for RO-SLAM, which is based on a SOG. This approach has revealed well-suited to the particular problems that arise in RO-SLAM, being significantly more efficient and accurate than other alternatives based on MC approximations. We have verified experimentally the robustness of the method against readings heavily corrupted with noise and outliers, as well as its ability to build 3-d maps from a real dataset gathered by UWB radio transceivers. Future research in this line will address the problem of the dependency on odometry by analyzing alternatives to the proposal density in the RBPF that do not rely on odometry. Another interesting topic is the evaluation of other strategies to reduce the number of SOG modes, e.g. enabling the fusion of Gaussian hypotheses.

CHAPTER 10

MEASURING UNCERTAINTY IN SLAM AND EXPLORATION

> "You should call it entropy, for two reasons. In the first place your uncertainty function has been used in statistical mechanics under that name, so it already has a name. In the second place, and more important, no one really knows what entropy really is, so in a debate you will always have the advantage."
>
> Scientific American 1971, vol. 225, page 180.
> John von Neumann

10.1 Introduction

As already made patent in previous chapters, Rao-Blackwellized Particle Filters (RBPFs) [Dou00a] have been intensively employed in recent years to address the SLAM problem [Gri07b, Mon02a]. In this scheme, probability densities are maintained by a set of weighted particles, each representing one hypothesis for the robot path. The Rao-Blackwellization consists of deriving maps analytically from these paths, which reduces

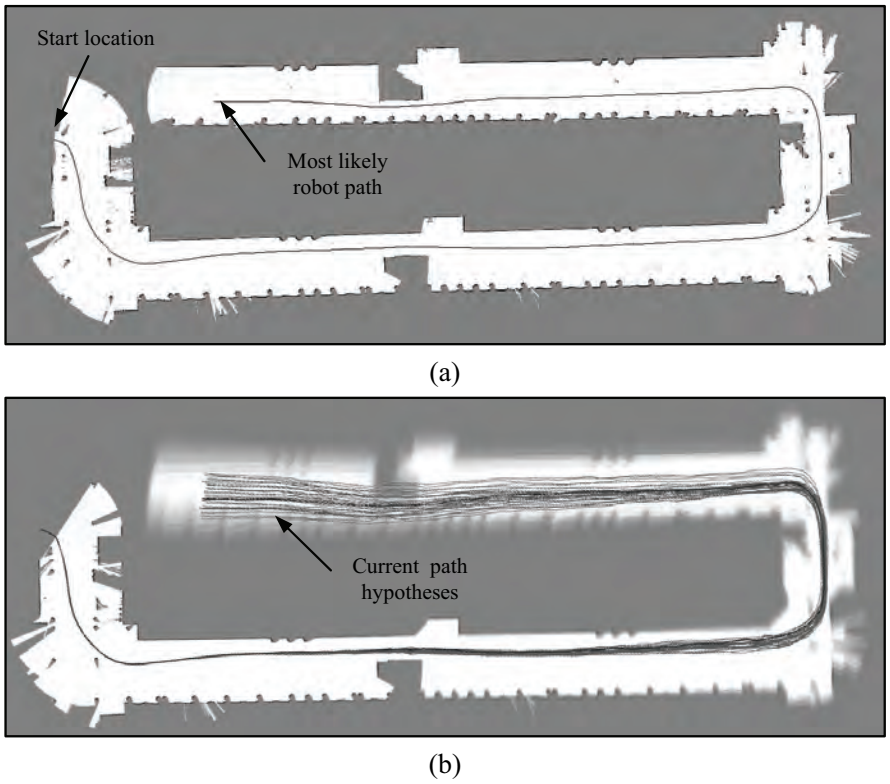


Figure 10.1: (a) The most likely map in RBPF mapping is the map associated to the particle with the highest weight. (b) In this work we introduce the expected map (EM), an average map where all particles are reflected in. The uncertainty of the RBPF in both the robot path and the map content can be effectively determined by this new map.

the dimensionality of the SLAM problem (recall Eq. 7.0.1). The most likely path (and therefore map) is usually considered to be the one associated to the particle with a highest weight, as the example in Figure 10.1(a).

In general, SLAM methods passively process incoming sensor data and iteratively update the estimates of the map and the path. However, the advantages of allowing the robot to actively control its movements while building a map, that is, *active exploration*, are well known and have been reported in the literature [Sim05, Sta04]. Exploration methods aim at controlling a robot in unknown scenarios in such a way that the whole environment is mapped while minimizing some cost function, such as the total traveled

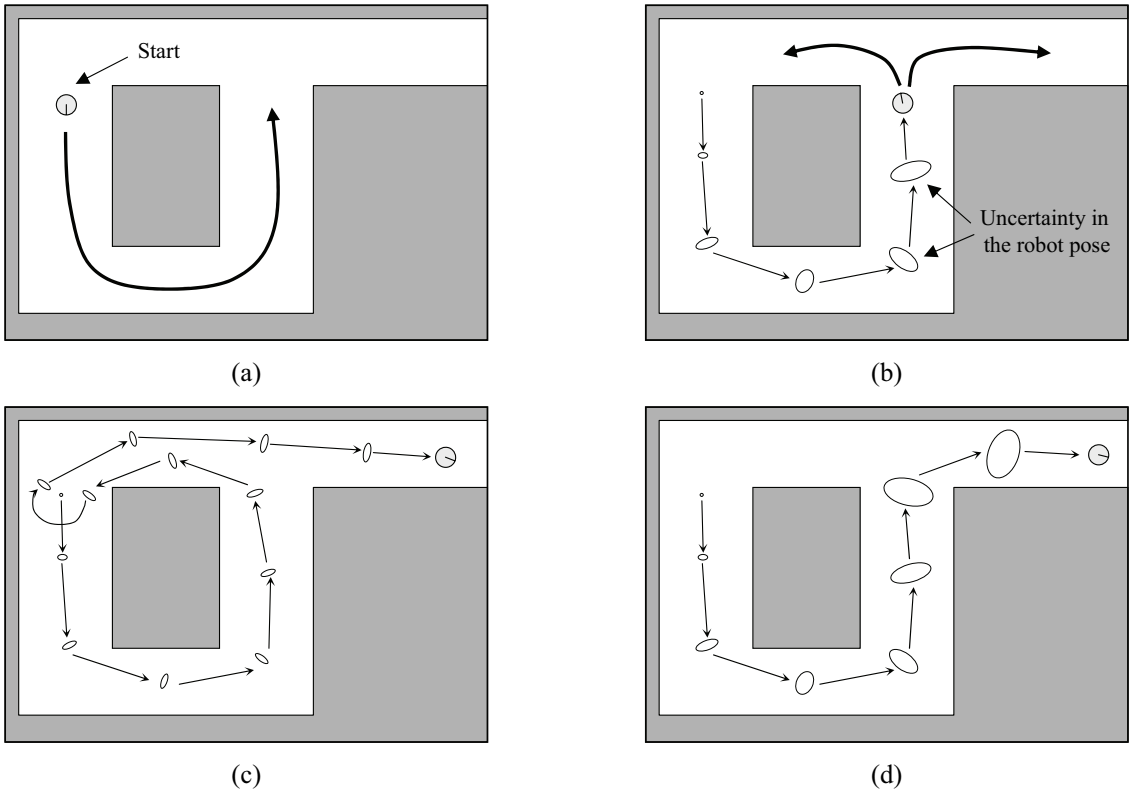


Figure 10.2: (a)–(b) The uncertainty in the robot pose while exploring an unknown environment increases as long as it does not revisit a known area, as schematically illustrated with uncertainty ellipses. If the robot explores a new corridor before closing the loop the resulting map will be much more inaccurate (d) than if it first closes the loop to reduce its uncertainty (c). Therefore, exploration methods should be favorable to perform loop closing.

distance [Bur00, Yam98] or the uncertainty in localization [Sta05b]. Regarding the latter goal, to illustrate the potential impact of movement selection in the accuracy of the resulting map, please consider the example in Figure 10.2(a) where a robot explores an environment with a loop (this example was inspired by a similar experiment reported in [Sta04]).

In Figure 10.2(b) the robot has traveled almost all the way round. The uncertainty in the localization along the path increases as long as the robot does not revisit a known area, where the registration between recent and past observations would reset the accumulated errors [Lu97a]. This is schematically illustrated in Figure 10.2(b) with

larger uncertainty ellipses for the robot pose as it moves further away from the origin. Next, the corridor at the right can be explored, but we should consider the convenience of previously closing the loop or not. In the first case, see Figure 10.2(c), revisiting the known area drastically reduces the localization uncertainty, thus the final map will be more precise than if the loop is not closed, which is the situation illustrated in Figure 10.2(d). Notice that higher uncertainty in the localization makes more problematic the detection of future loop closures. Therefore, the mapping process will be easier if the robot revisits known places as soon as possible to reduce its localization uncertainty.

Apart from the uncertainty in the robot path, uncertainty also exists in the maps due to the lack of knowledge about unexplored areas, the noisy nature of sensors and the misalignment of observations taken from poses poorly localized. Thus, any integrated approach considering SLAM and active exploration must take into account both uncertainty sources when deciding movement actions: on the one hand, it is desirable to take the robot towards unknown places in order to incorporate new information into the map, but on the other hand this will usually decrease accuracy in the localization (until revisit). Approaches in the literature quantify only one of these opposed factors, both of them at a time, or propose some kind of combined measurements [Bou02, Sta05a]. The main concern in this chapter is that measuring the uncertainty of an RBPF is a fundamental part of the techniques for mapping environments with multiple nested loops [Sta05b, Sta04].

Therefore, in this chapter we introduce a new uncertainty measure that simultaneously considers the robot path and the map content. The key idea behind this method is that of evaluating the consistency between individual maps from all the hypotheses (particles) in an RBPF: since these maps are generated from each path hypothesis we are also implicitly evaluating the consistency between the estimated paths. Recall that

RBPFs address the full SLAM problem (recall Chapter 7) where each sample keep the whole estimated path so it is able to reconstruct the associated metric map.

With the purpose of evaluating the consistency between hypotheses we introduce the concept of the *expected map* (EM), a weighted average of maps computed from the contribution of all the particles in the RBPF. As can be observed with the example in Figure 10.1(b), inconsistencies between maps appear in an EM as blurred areas. Instead of measuring the uncertainty in this map directly through its *entropy* (the natural measure of uncertainty), it will be defined defined here the *information* (I) and the *mean information* (MI) of a grid map. Both metrics are based on the entropy but avoid some of its shortcomings when applied to grid maps. In particular, the presented measures are independent of the grid map extent, while the MI is practically independent of the map resolution for typical grid cell sizes too. Although both MI and I can be applied to grid maps in general and not only to RBPF-based SLAM, they will be analyzed here in the context of obtaining a measure of the uncertainty in the EM of a given RBPF.

In summary, two different uncertainty measures are introduced in this chapter:

- The mean information of the EM (called here EMMI) provides an alternative to the entropy of the robot path in applications such as loop closure detection [Sta04]. As will be discussed later on, the entropy of the path may not become well-defined in the context of an RBPF due to the sparsity of the particle representation. This does not affect EMMI.

- The information of the EM (EMI) performs more appropriately than other measures (including EMMI) for the case of exploration using occupancy grid maps, since actions aimed at closing loops are given more importance against the

exploration of new areas when the current pose uncertainty is high. Benefits of closing loops while exploring have been reported by other authors elsewhere [Sta05b, Sta04].

It must be remarked that the presented methods are *not* a different way of computing the joint entropy of an RBPF (defined below in §10.5.1), which has a unique, well-defined mathematical definition. Instead, we propose an alternative way of measuring the uncertainty in a RBPF by means of the entropy of a new variable (the expected map) in order to avoid the undesirable property of the joint entropy of missing the opportunity to close loops in grid map-based exploration.

The organization of the rest of this chapter is now sketched. Firstly, we present a review of previously-proposed uncertainty measures. Then, the EM is defined in the general framework of SLAM and in the concrete case of an RBPF. The new information metrics for grid maps is then presented in §10.4, and next it is compared to other uncertainty measures. Finally, §emi:sect:Results contains some experimental results for active exploration and loop-closure detection using the proposed measures.

10.2 Existing uncertainty measures in SLAM

Probabilistic approaches dominate the research in robot localization, map building and active exploration. Classically, the most widely employed probabilistic representation for robot poses has been the multivariate Gaussian distribution, where both a mean pose vector and a covariance matrix are maintained [Dis01]. This approach can be used in the contexts of position tracking (where the initial pose distribution is known), global localization (there is no knowledge about the robot starting location) and landmarks-based SLAM (both the pose of the robot and a set of landmarks are to be estimated).

This latter perspective has been successfully addressed by Kalman or Extended Kalman Filtering (KF [Kal60] and EKF [Jul97], respectively).

Uncertainty in these filters is related to their covariance matrices, thus the entropy of such matrices becomes a natural uncertainty measurement [Bou02, Vla99]. However, (i) the intractable growth in complexity of these filters of $O(N^2)$ for N landmarks, (ii) their restrictive assumption of uni-modal Gaussian distributions and (iii) their inability to deal with raw data (features must be firstly extracted), have led to the proposal of more efficient approaches in the last years.

In this sense, particle filters have gained a huge popularity in the mainstream robotic research since the usage of a limited number of particles assure a bounded computation complexity. In the contexts of SLAM and active exploration, RBPFs represent a very efficient solution, allowing us to simultaneously estimating both the robot poses (path) and the map [Dou00a, Sta05a].

There are some proposed measurements for the uncertainty of the robot pose only (ignoring the map), such as the volume covered by particles [Sta04] or the entropy of the Gaussian distribution which approximates the particles [Sta05a]. A particularly original proposal can be found in [Fox03], where the Kullback-Leibler distance (KLD) is used to measure (and bound) the error caused by the approximation of the pose distribution by a discrete set of particles.

Regarding the uncertainty in maps, entropy has also been employed as a measure for different map representations: point-clouds, landmarks, and occupancy grids [Bou02, Sae05]. However, in RBPF-mapping we have multiple map hypotheses simultaneously. Thus, the entropy of maps needs a proper mechanism to explicitly consider the consistency between hypotheses, a role played by the EM in the approach presented in this chapter and not found, to the best of our knowledge, in any previous work. Although entropy is a founded measure of the information in maps, its direct

use on these data has some important drawbacks, as exposed in the next section.

The methods mentioned above consider the uncertainty in either the robot pose or the map separately. Others have proposed combined estimators which simultaneously take into account both sources of uncertainty: it seems reasonable to think that taking into account just one uncertainty source is inappropriate for active exploration if the goal is obtaining both a good localization and a consistent map. In the work reported in [Bou02], both entropies are computed separately and then weighted together, with weights obtained experimentally. A more elegant method is reported in [Sta05a], which computes the joint entropy of the two variables. In spite of this latter method being mathematically founded, a number of problems discussed later on limits its utility in practice. Interestingly, drawbacks to the usage of joint entropy as the basis for an exploration policy are also discussed in [Sim05], where an alternative uncertainty measure is presented for EKF-based exploration.

In the context of multi-robot exploration, Ko *et al.* proposed in [Ko03] two different heuristic functions for the cases of exploring a new area and meeting another robot (an event equivalent to a loop closure), while in [Bur00] other non-entropy based utility functions are proposed to prevent different robots to explore the same areas.

From the methods discussed above, the joint entropy is used throughout this chapter as a reference for comparisons, since it is one of the most significant in the context of RBPF-based grid mapping.

10.3 The expected-map (EM) of an RBPF

This section firstly sets out the employed mathematical variables and notation. Next, we introduce the concept of the expected map (EM) for an RBPF and discuss its properties.

10.3.1 Preliminary definitions

Assume a robot moving in a planar scenario whose location at time t can be described by a 2-d pose x_t. The real pose at each instant of time is unknown, but can be estimated through sequential Bayesian filtering. Within an RBPF [Dou00a], we estimate the probability density of the robot path $x^t = \{x_1, ..., x^t\}$ through a particle filter that implements recursive Bayesian filtering, as explained in Chapter 7. For convenience of the reader, the final expression for the estimation of the robot path is repeated here:

$$p(x^t|z^t, u^t) \propto p(z_t|x^t, m^{[i]}) \int \cdots \int_{-\infty}^{\infty} p(x_t|x_{t-1}, u_t) p(x^{t-1}|z^{t-1}, u^{t-1}) dx^{t-1} \quad (10.3.1)$$

where the z_t are sensor observations, and the u_t are robot actions (typically incremental displacements measured by odometry [Del99] or incremental range scan matching [Hah03, Sta05b]).

One of the most popular representations for maps in the robotics community are occupancy grids, widely employed during the last twenty years [Elf89, Gri07b, Mor88, Mor85, Sta04, Thr03]. An occupancy grid map m is a discrete random field where we store the occupancy probability for each cell, which we will denote as $p(m_{xy})$ for a cell with indexes $\langle x, y \rangle$. Under no prior information available about obstacles, the occupancy probability for all cells can be initially set to 0.5: each cell can be either occupied or free with the same probability. A grid map is updated by integrating sensor

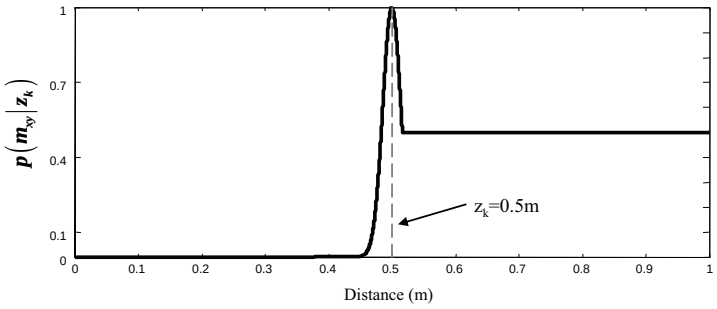


Figure 10.3: The probabilistic inverse model of a sensor estimates the occupancy of map cells given that some reading z_k (distance to obstacle) has been obtained. In the figure this density is plotted for a laser range finder (and for a measurement of $z_k = 0.5m$) as a function of the distance from the sensor to cells.

measurements through the so called *inverse sensor model* [Thr05],

$$p(m_{xy}|x_t, z_t) \tag{10.3.2}$$

that is, the likelihood of the cell m_{xy} being occupied, conditioned on a given observation z_t taken from a pose x_t. Fusion of the current observation with the previous contents of the map can be done on a Bayesian basis by using the following iterative expression:

$$p(m_{xy}|x^t, z^t) = \left(1 + \frac{1 - p\left(m_{xy}|\, x^{t-1}, z^{t-1}\right)}{p\left(m_{xy}|\, x^{t-1}, z^{t-1}\right)} \cdot \frac{1 - p\left(m_{xy}|\, x^t, z_t\right)}{p\left(m_{xy}|\, x^t, z_t\right)}\right)^{-1} \tag{10.3.3}$$

which can be easily derived from the log-odds representation of the update process [Mor88] if we assume an initial occupancy likelihood of 0.5 for all cells. Essentially, Eq. (10.3.3) increases the certainty in the occupancy/freeness of a given cell if subsequent observations confirm the current belief. The only density required to iterate Eq. (10.3.3) is the inverse sensor model in Eq. (10.3.2). For the common case of a laser range scanner we can consider a function like the one depicted in Figure 10.3.

At this point we have defined stochastic representations for both the robot path and the map (in Eq. (10.3.1) and Eq. (10.3.3), respectively), thus we could compute

their entropy values separately to measure their uncertainty. We propose instead to build a new map, the EM, aimed both at revealing inconsistencies between the map hypotheses in an RBPF and at measuring the overall uncertainty of the filter.

10.3.2 Definition of the expected map

The expected map (EM) is defined as the mathematical expectation of the probability distribution of the map taken over all the possible paths, given by the posterior $p(x^t|z^t, u^t)$, that is, the result of marginalizing out the robot path from the SLAM posterior:

$$p(EM|z^t, u^t) \doteq \mathrm{E}_{x^t}\left[p(m|x^t, z^t, u^t)\right] \tag{10.3.4}$$

In the context of SLAM we know the posterior of the robot path, given by Eq. (10.3.1). Thus, the definition of the EM above can be rewritten as:

$$p(EM|z^t, u^t) = \int \cdots \int_{-\infty}^{\infty} p(m|x^t, z^t, u^t) p(x^t|z^t, u^t) dx^t \tag{10.3.5}$$

In the concrete case of RBPF-based SLAM and occupancy grid maps, the EM is also a grid map where the occupancy of each cell EM_{xy} can be obtained from the contributions of all the map hypotheses in the RBPF, each one associated to a path hypothesis, transforming the above integral into a sum:

$$p(EM_{xy}|z^t, u^t) \approx \sum_{i=1}^{M} \omega_k^{[i]} p(m_{xy}|x^{[i],t}, z^t, u^t) \tag{10.3.6}$$

where M is the number of particles in the filter, and the $\omega_k^{[i]}$ are their associated importance weights. The intuition behind Eq. (10.3.4)–(10.3.6) is that by contrasting the

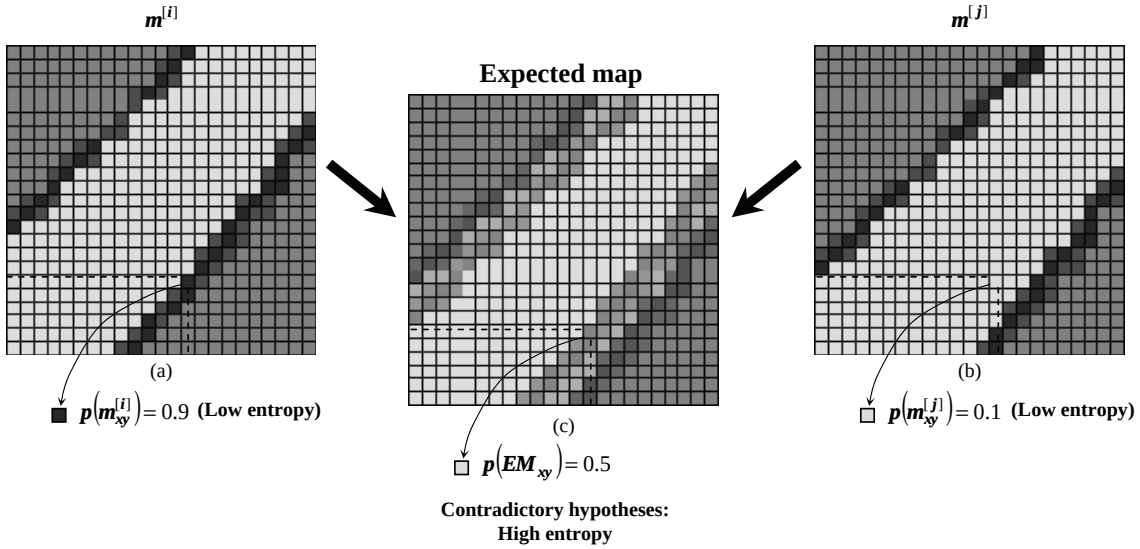


Figure 10.4: (a)–(b) Map hypotheses according to a pair of particles i and j, respectively. (c) Their combination into an expected map (EM) reveals contradictory values for some cells (like the one highlighted in the figure) by means of a value around 0.5 that reflects a high occupancy uncertainty.

occupancy values of cells m_{xy} at the same location (the same $\langle x, y \rangle$ indexes), but from maps associated to different particles, we can test the coherence between hypotheses, which can not be measured directly by other methods like the joint entropy (as shown in section 10.5.2). The resulting map can be visualized as an image where sharp areas indicate information that is certain, whereas blurred regions involve uncertainty, i.e. there are contradictory hypotheses about their content. This is illustrated in Figure 10.4(c), where we obtain an EM with uncertain occupancy values (close to 0.5) for cells with contradictory content in the individual maps. Another example for real data has been shown in Figure 10.1(b).

In the next section we define information metrics capable of measuring the information into an EM. As expected, those measures assign a higher amount of information to "sharp" than to "blurred" maps. If we think of the EM in an active exploration

framework, it is clear that desirable actions are those ones leading to a gain of information in the EM, caused either by the exploration of new areas or by the closure of a long loop.

10.4 Information measures for grid maps

In the following we define two metrics which measure the information in a given occupancy grid map that we will apply, in particular, to EMs. Information theory establishes that the information associated to a random variable is related to its entropy [Cov91]. Since each grid cell is a discrete random variable with two possible outcomes, i.e. a Bernoulli distribution, its entropy is given by:

$$H(m_{xy}) = -p(m_{xy}) \log p(m_{xy}) - \bar{p}(m_{xy}) \log \bar{p}(m_{xy}) \tag{10.4.1}$$

with $\bar{p}(m_{xy}) = 1 - p(m_{xy})$. Notice that the maximum attainable entropy is given for $p(m_{xy}) = 0.5$, that is, for unobserved cells. Assuming statistical independence between cells, we can compute the entropy of the whole map m as:

$$H(m) = \sum_{\forall x,y} H(m_{xy}) \tag{10.4.2}$$

These equations for computing the entropy of a grid map have been widely used as a measure of the information in maps [Bou02, Sta05a]. However, in practice it exhibits the following drawbacks:

- Its absolute numerical value depends on the grid extent (the rectangular limits of the map) instead of the actual observed area. Notice that Eq. (10.4.2) implies that

all the unobserved cells in a map contribute to the entropy with their maximum value of uncertainty.

- It also depends on the grid resolution, since that parameter settles (along with the map extent) the total number of cells in the map. This means that the entropy of any map with unobserved areas (all maps in practice) increases without bounds when resolution increases. We must remark that corrective scale factors have been proposed elsewhere to alleviate this problem [Sta06].

In order to overcome these drawbacks we define here the information (I) of a map m as the following entropy-based measure:

$$\begin{aligned} I(m) &= \sum_{\forall x,y} I(m_{xy}) \qquad \text{(bits)} \\ I(m_{xy}) &= 1 - H(m_{xy}) \end{aligned} \tag{10.4.3}$$

where for convenience the entropy $H(\cdot)$ is computed using base-2 logarithms. As a result we obtain a natural measure of information in units of bits. Noticeably, the maximum information value (1 bit) for $I(m_{xy})$ is given to a certainly occupied/free cell, while the minimum value (0 bits) is associated to any unobserved cell. It is evident that therefore the dependency of the entropy on the grid extent is avoided using this definition of information: the limits of the map become irrelevant since all the unobserved cells now contribute with a null information. Thus $I(m)$ is a more practical and normalized quantifier of the uncertainty in a map than the direct application of the entropy.

In addition, the *certainty* of the content of a whole map m can be measured by the mean information (MI), or $\bar{I}(m)$, defined as

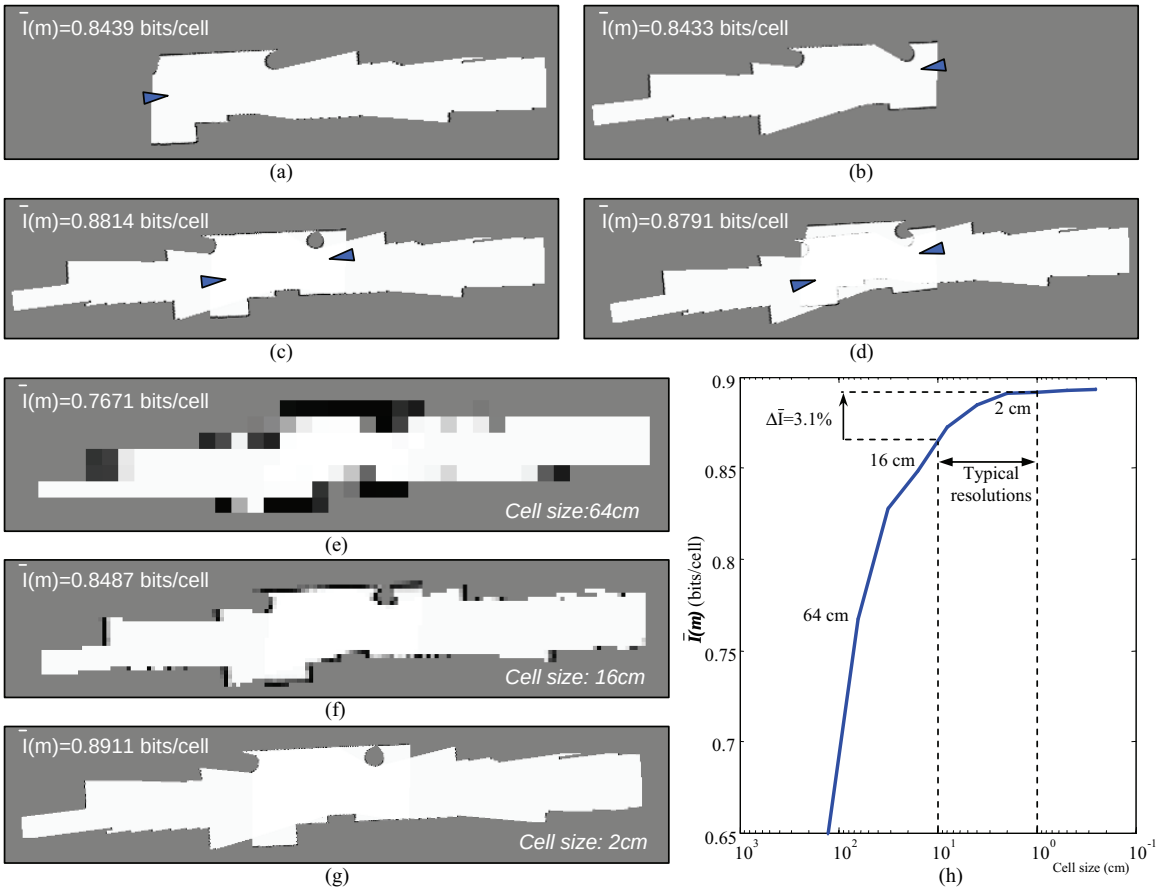


Figure 10.5: Examples illustrating the main properties of the MI ($\bar{I}$).(a)–(b) Two observations and the MI value of the maps built from them separately. (c) When both are aligned and fused in the same grid, the information value increases. (d) Inconsistencies due to poor robot localization decrease the quality of the fused map, which is confirmed by a lower MI value. In (e)–(g) the same map is shown for different cell resolutions, along with their MI values. The MI values obtained for this environment are plotted in (h) for a wide range of resolutions, from 128cm to 0.25cm. Observe the small change in the MI value when varying the resolution of a map from 1cm to 10cm, which coincides with commonly used resolutions. Thus, in practice, the MI can be regarded as resolution-independent for maps with cell size smaller than 10cm.

$$\bar{I}(m) = \begin{cases} I(m)/N_{obs} & , \text{ if } N_{obs} > 0 \\ 0 & , \text{ otherwise} \end{cases} \qquad \text{(bits/cell)} \tag{10.4.4}$$

where N_{obs} represents the number of observed cells in the map, i.e. all those cells with occupancy likelihood different from 0.5. This scale factor is aimed at bounding the value of MI to the range $[0, 1]$, whereas the straightforward computation of entropy gives values that can increase without bound.

Consider the following example, which illustrates a key feature of the MI. Two observations are captured from different poses within the same environment, which separately give rise to the maps shown in Figure 10.5(a)–(b). Alternatively, Figure 10.5(c) shows an occupancy grid where both observations are fused by means of Eq. (10.3.3) using the correct alignment. Here, each observation confirms the other one, thus occupancy values of cells are closer to 0 and 1 than in the previous maps made from a single observation. Consequently, the MI of this map is greater than before since we have more certain information. This behavior follows from the properties of the information of a map as defined in Eq. (10.4.3), whose maximum value is obtained for occupancy values of 0 and 1. To illustrate an opposed situation, please notice how both observations are misaligned in Figure 10.5(d), which is given a lower MI value.

To sum up, we enumerate next the properties of our information metrics I and MI that set them apart from the direct application of entropy to grid maps:

- An empty map (containing only unobserved cells) has a null information and a null mean information.
- Both measures are independent of the grid map rectangular limits, since unobserved cells do not contribute to the information in the map.
- The MI is mostly independent of the grid resolution for practical cell sizes. This is

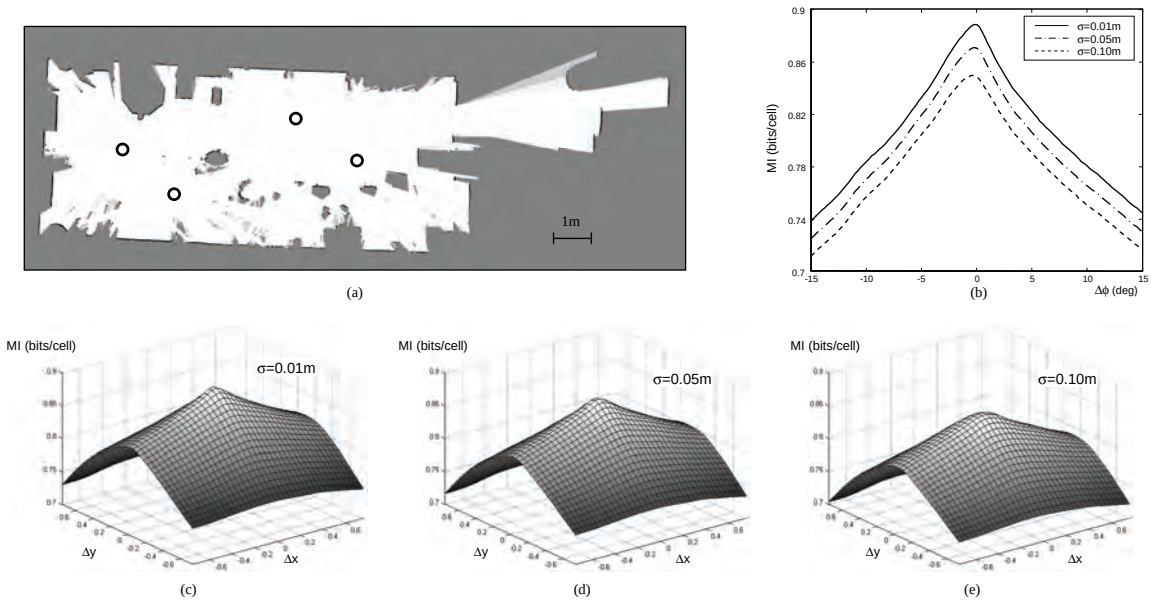


Figure 10.6: (a) A synthetic environment used to test the MI response against sensor noise and localization errors. Observations have been simulated from the circled positions, a map has been generated from them and then its MI computed. (b) The results for orientation errors only and different levels of sensor noise. (c)–(e) The response of the MI against errors in positioning (with the correct orientation). It can be noticed the higher selectivity for less noisy measurements. Anyway, the maximum value is obtained in all cases for the correct position and orientation, thus the results show the ability of the MI to reflect the consistency between different observations.

shown in the maps of Figure 10.5(e)–(g), whose MI increases as we consider higher resolutions. However, the MI asymptotically tends towards a maximum value, as can be appreciated in Figure 10.5(h). This presents a remarkable difference with the performance of the entropy, which in that case tends to infinite. The asymptotic behavior of MI depends on the inverse sensor model, the specific environment being mapped, and other factors. In appendix D we show how a closed form expression can be derived for a given synthetic environment.

- The better the alignment between observations in the map, the higher the values obtained for the MI, as can be observed in Figure 10.5(c)–(d). The intuitive idea behind this property is that well-aligned observations make the occupancy of cells

to tend toward either 0 or 1, which correspond to maximum information values. This property is the key for the distinctive behavior of the EMMI uncertainty estimator when closing a loop, as shown later on.

We discuss next the experiment summarized in Figure 10.6, which is aimed at showing how we obtain high MI values for maps built from well aligned observations (the last property from the list above). Simulation has been chosen here for knowing the real robot pose and having the possibility of changing sensor parameters freely. We have computed several sets of simulated observations (laser range scans) from the surroundings of the circled positions marked in Figure 10.6(a), according to some given errors in positioning (x, y) and orientation (θ). The observations are also corrupted with an additive Gaussian noise of standard deviation σ. Next we compute the MI of the grid map built from these observations following Eq. (10.3.3). The average results are plotted separately in Figure 10.6(b)–(e) for errors in orientation and position, and for different levels of sensor noise. As expected, the highest MI value is obtained for the correctly aligned observations. We can also remark how more precise observations (with lower error σ) lead to more selective MI values.

10.5 Comparison to other uncertainty measurements

In the following we compare the proposed uncertainty measures, namely the information and mean information of the EM (EMI and EMMI, respectively) to other entropy-based methods, such as the entropy of the robot path [Bur97, Roy99, Vla99], the effective sample size (N_{eff}) [Liu96], and the path-map joint entropy [Sta05a]. We provide here a theoretical discussion about their complexities and their expected behaviors in typical mapping situations, while experimental results supporting our proposal

are shown in the next section.

10.5.1 Expected behaviors for the uncertainty measures

Before comparing the behavior of the different measures, it is convenient to consider the expression of the joint entropy applied to RBPF [Sta05a], the method most related to ours, to better understand its properties:

$$\begin{aligned} H(x^t, m|z^t, u^t) &= H(x^t|z^t, u^t) + \int_{\infty}^{\infty} p(x^t|z^t, u^t) H(m|x^t, z^t, u^t) dx^t \quad (10.5.1) \\ &\approx H(x^t|z^t, u^t) + \sum_{i=1}^{M} \omega_k^{[i]} H(m|x^{[i],t}, z^t, u^t) \end{aligned}$$

It is clear that this measure is a composition of the entropy of the robot path and the average entropy of individual maps $p(m|x^{[i],t}, z^t, u^t)$ weighted by $\omega_k^{[i]}$ for each particle i. Three fundamental drawbacks can be pointed out from Eq. (10.5.1):

- The term that considers the entropy of the maps (the integral in Eq. (10.5.1)) dominates the overall value, hiding therefore the contribution of the path uncertainty (the first term in that expression). The reason is that the former is typically many orders of magnitude higher than the entropy of the path. In our experiments, we have found a typical ratio between both terms in the order of 10^5.

- Only the content of individual maps determines the result, independently of their mutual consistency: this method is unable to detect inconsistencies between particles. In other words, the joint entropy is largely determined by the number of observed cells in maps, being mostly independent of whether individual maps contradict each other.

- It is not clear how to compute the entropy of the robot path (the first term in Eq. (10.5.1)) in an RBPF. We will consider here a good approximation of this value based on fitting Gaussians to the robot pose at each time step and averaging their entropy along the whole path, as proposed in [Roy99, Sta05a]. However, we will also show below that this value can become undefined after closing a loop.

We now examine the theoretical behavior of the four uncertainty measures (EMI, EMMI, $Neff$ and joint entropy) for two distinctive situations that can occur while performing mapping or exploration with RBPFs. The purpose of this comparison is to show how the EMI and EMMI estimators represent a more reliable measure than the others for selecting robot actions and detecting loop-closures, respectively.

Exploration of new areas

In this case, particles tend to spread in space while their weights remain practically constant. As a result, the N_{eff} of the RBPF remains practically constant, while the entropy of the path raises due to the higher uncertainty in localization. Note that this entropy is one of the components of the joint entropy, but in the case of exploring new areas the increase in the grid map information by far compensates the increase of uncertainty in the robot path, thus the overall value of the joint entropy raises while exploring. Regarding our uncertainty measures, the information in the EM (the EMI) increases too, but the number of observed cells grows as well. However, the increase in the robot pose uncertainty continuously introduces small incoherences between individual maps, which is reflected by decreasing EMMI values. Therefore, EMMI is worse suited than EMI for detecting the increase in information that occurs when exploring new terrain.

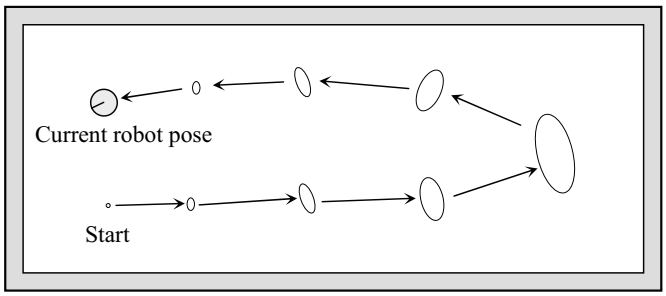


Figure 10.7: An example of how the uncertainty in the robot path decreases when it goes back traversing already known areas only. However, closing a loop implies a much more drastic reduction of the uncertainty than going back through a known path. As discussed in the text, our uncertainty measures clearly reflects the difference between both situations.

Loop closure

In SLAM, returning to an already known place through an unknown path is a special case of revisit called loop closure – the other option for revisit (through an already explored path) is illustrated in Figure 10.7. If the closed loop is long enough, typically only a few particles will be close to the actual path. This will result in: (i) only a few particles with non-negligible weights, (ii) particles will be resampled, and (iii) the pose uncertainty will be largely reduced [Aru02, Gri07b]. It is clear here that both N_{eff} and the robot path entropy fall drastically. Since the path entropy is also a term of the joint entropy (and the other term for the map contents remains practically constant in this situation), we can conclude that the joint entropy also falls when closing a loop.

In practice both path and map entropies above break down due to the particle approximation of the SLAM posterior. The reason is that typically only one hypothesis survives to the resampling (print (ii) above) after closing a long loop (for example, refer to [Sta04]). Thus, the covariance of the Gaussians fitted to the surviving robot path estimate collapses to zero. Since the entropy of a d dimensional Gaussian $x \sim \mathcal{N}(\mu, \Sigma)$ is given by:

$$H(x) = \frac{1}{2}\log((2\pi e)^d|\Sigma|) \tag{10.5.2}$$

this would imply a minus infinity entropy since $|\Sigma|$ tends to zero – for instance, a standard deviation of 1mm in x and y has an associated entropy of -10.98.

We believe that one solution to this numerical problem is to impose a lower limit to the covariance of the robot pose, although in that case, the heuristically-chosen value strongly determines the behavior of the joint entropy when applied to exploration. We must highlight the small range of values for the path entropy in comparison to the second term of the joint entropy (the uncertainty in maps), which makes that, in practice, the content of maps determines the overall measurement of the uncertainty, disregarding the uncertainty in the path.

Consider the following numerical example, which illustrates the problem of using the joint entropy for choosing actions in exploration. A robot faces two potential actions, which are evaluated trough the joint entropy. The first action closes a long loop and reduces the robot pose uncertainty in x and y from $1m$ to $1mm$ (considering these values as the standard deviations), whereas we predict that the second action will allow us to explore a new area of $1m^2$ (100 grid cells for a $0.10m$ cell size). Straightforward calculations [1] give us an expected reduction of the joint entropy of 13.82 and 69.3 for the first and the second action, respectively. Obviously, closing the long loop is a much more desirable action for the robot in that case than exploring a small area of new terrain [Sim05, Sta04], thus the joint-entropy would not be a good choice for active exploration.

[1] First case: pose uncertainty is reduced. We have to evaluate Eq. (10.5.2) for 2-d Gaussians with $\sigma = 1$ and $\sigma = 0.001$, then subtract the so-obtained entropies to obtain the information gain $\Delta H_1 = -(2.84 - (-10.98)) = -13.82$ nats. Second case: grid cells are observed. The maximum entropy reduction per grid cell is 0.693 nats, thus for 100 cells we have $\Delta H_2 = -69.3$ nats. These quantities were computed using natural logarithms instead of base-2 ones, but this fact is irrelevant for the present argument. Note that *nats* are the units of information for natural logarithms.

In contrast, the EMI of the RBPF after closing a loop always increases due to the elimination of incoherent hypotheses (which in turn generate more "blurred" EMs), the same reason that also makes the EMMI to raise. The particle depletion problem does not affect our measures since the path hypotheses are taken into account through the EM, which is always well defined.

10.5.2 Consistency detection between map hypotheses

We discuss now a key difference between how the EMMI and the joint path-map entropy detect inconsistencies between the individual maps from each particle in an RBPF. In the joint entropy, the entropy of the map contents $H(m)$ contributes to the total value averaged by the weights of the associated particles. In contrast, the EMMI considers the entropy of the cells in the EM, which, in turn, are the weighted average of the maps from each particle. Consequently, the EMMI computes the entropy of the average of maps, whereas the joint entropy performs these operations *in the opposite order*.

To graphically see the important implications of this difference, Figure 10.8(a)–(b) illustrate how the joint entropy fails to capture inconsistencies. Here, the graphs show the entropy terms involved in each method for two equally probable particles, each holding a hypothesis for the state of just one single cell of a grid map (these simplifications have been taken just to enable the visualization of the results in 3-d). The key observation is that inconsistencies such as one hypothesis stating that the cell is free (a value near 0) and the other stating it is occupied (a value near 1) are detected by EMMI as uncertainty, that is, a high entropy value (see Figure 10.8(b)). In contrast, similar situations are assigned a low entropy value (high certainty) by the joint entropy, as can be observed in Figure 10.8(a).

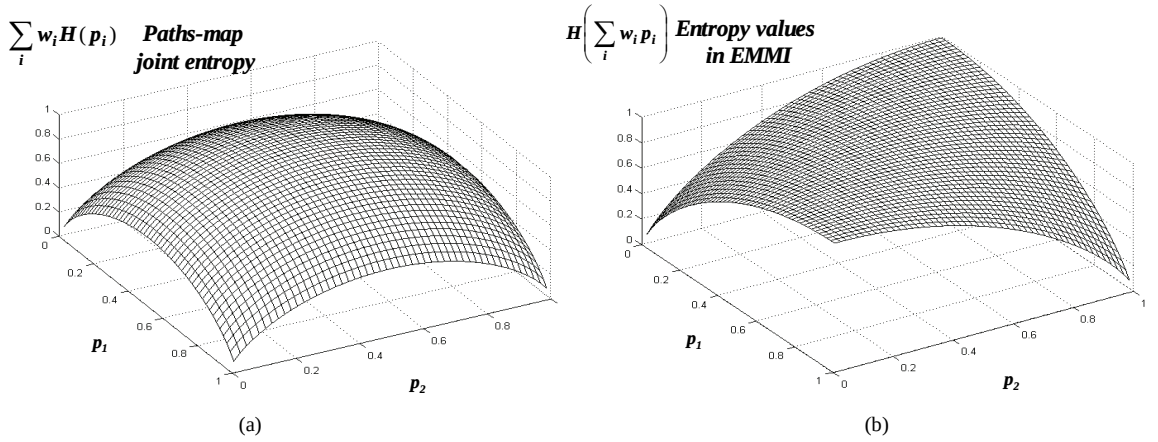


Figure 10.8: Entropy terms involved in the paths-map joint entropy (a) and in the EMMI (b) are plotted for the case of only two particles, as an example of how both approaches deal with different hypotheses. It is clear that the first one considers as certain (low entropy) contradictory values, like $p_1 = 0$ (free cell) and $p_2 = 1$ (occupied cell), whereas EMMI only gives low entropy values when both likelihoods are alike (consistent).

10.5.3 Computational and storage complexity discussion

Regarding the computational time complexity of all these uncertainty measures, the most efficient ones are the effective sample size N_{eff}, with O(M), and the entropy of the robot path, which involves O(ML) operations, being M and L the number of particles and the length of the path, respectively. On the other hand, the joint entropy has a complexity $O(M(N + L))$, where N is the number of cells in the grid. Clearly, N will be the dominant quantity in practice, thus this complexity will tend toward $O(MN)$. Since each observation will modify only a limited area of the grid maps, we can reduce even more the computation complexity of the joint entropy to $O(M)$.

Unfortunately, the construction of the EM (required for the evaluation of the EMI and EMMI measures) depends on the weights of the particle filter, hence it cannot be simplified and the complexity of building the EM is always $O(MN)$. This is the cost to pay for detecting the consistency between different map hypotheses.

Concerning the storage demands, the methods presented in this chapter require the

computation of one additional map, the EM, apart from the M maps associated to the particles in the RBPF, whereas the joint entropy does not require this supplementary storage. However, this cost is not significant since it implies keeping only $M+1$ maps instead of M, for M being the number of particles.

10.6 Experimental evaluation and discussion

In this section we provide an experimental validation of the presented uncertainty measures through comparisons with other reported alternatives. Firstly, we perform a statistical analysis of the EMMI measure in order to check its ability to detect loop closures through the changes in the EM of the RBPF. Next we show how EMI can be applied to active exploration and compare its performance to that of the joint path-map entropy.

10.6.1 Characterization of loop closing detection with EMMI

Due to the stochastic nature of particle filters, it is necessary to carry out a number of realizations of each RBPF simulation in order to obtain statistically significant results. This is the reason why in the following experiment we have repeated the mapping process 20 times (each realization takes 20 minutes in a 3.2GHz Pentium 4), leading to a convenient comparison of the different uncertainty measures by means of their mean values and variances. The experiment consists of an optimized RBPF [Gri07a] which consumes data gathered by our robotic wheelchair SENA [Gon06a, Gon06b] in the streets surrounding a building in our campus (see Figure 10.9). Sensory data consist of odometry readings and scans gathered by a SICK laser range finder.

In this experiment the robot completes two laps around the building, a total traveled

distance of 550m, at an average speed of 1.5m/sec. The most critical point is when it closes the loop for the first time. Figures 10.9(a)–(d) represent the evolution of the EM at some instants of time during one realization of the map-building process. The "blurred" aspect of the EM is more perceptible as the robot gets farther along the loop, until it definitively closes it. This closing approximately starts at time step 150, where the uncertainty due to the particles dispersion starts to decrease. After that, the path estimation and the map remain accurate while the robot travels again over the same path. This illustrates the fact that the uncertainty continuously increases while the robot explores unknown areas, and that it rapidly decreases when the loop is closed. To measure these changes we have computed both the EMMI and the joint entropy (theoretically compared in section 10.5). Additionally, the effective sample size N_{eff} and the $3\sigma(99.7\%)$ confidence area occupied by the particles in the space have also been computed for illustration purposes, although they are not founded uncertainty estimators. Statistical results are plotted in Figure 10.9(e)–(h), where mean values from the 20 experiments are shown in solid lines, and $\pm 2\sigma$ confidence intervals appear dashed. Note that these intervals are for fitted Gaussian distributions, hence they can exceed the valid range of the variables, e.g. the upper confidence limit in Figure 10.9(g) goes above 20 which is the maximum value, or the lower confidence limit in Figure 10.9(h) which gets negative.

The evolution of the uncertainty estimators can be interpreted as follows. In the case of the EMMI, it decreases (less certainty) while the robot travels along the loop for the first time. After the loop closure, this estimator restores its high value (more certainty). This behavior can be observed in Figure 10.9(e) in the first gradual decrease and the posterior rise beyond time step 150. Notice how the EMMI remains practically constant during the rest of the experiment. On the other hand, the joint entropy has shown to be the estimator with the smallest variance, i.e. its values do not vary significantly

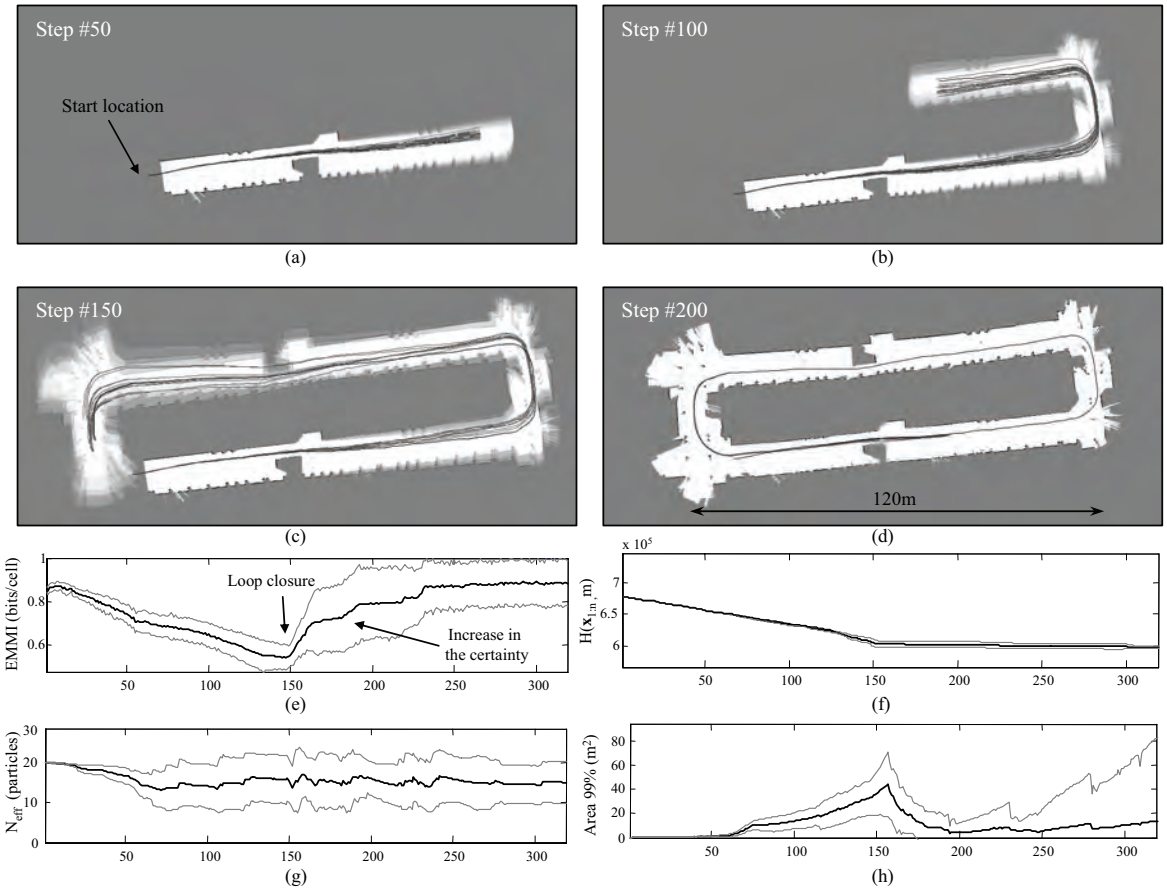


Figure 10.9: The EM for some steps of the mapping process can be seen in (a)–(d). The computed uncertainty estimators are the EMMI (e), the joint entropy (f), the effective sample size (g), and the area occupied by the particles (h). The EMMI shows its ability to better reflect the actual uncertainty of the RBPF than the others. These plots are the results of 20 realizations of the same mapping process. The mean values for all the estimators are plotted as solid lines together with their 95.4% confidence intervals ($\pm 2\sigma$ using approximate Gaussian fits) in dashed.

between different runs. This is due to the dominance of the maps entropy in the estimator, which only depends on the number of observed cells, being (practically) independent of inconsistencies between hypotheses. Consequently, from the results given by this measure, shown in Figure 10.9(f), we can only know that from step 150 to the end of the experiment, a small number of cells has been updated in the maps. Notice that if we would use this estimator in active exploration to decide robot actions, we could hardly distinguish between closing a loop or any other action that does not involve exploration of a new area, e.g. going back along the traversed path, remaining still,... This contrasts with the competent performance of the EMMI in this respect. Regarding the results for the effective sample size (N_{eff}), it can occasionally reflect a loop closure by means of restoring its maximum value due to an associated particles resample. However, the exact moment at which a resample occurs depends on many implementation parameters. Moreover, a resample can occur many times after a loop closure, or even while exploring new areas, as previously discussed. All these facts are reflected in the large variance observed in Figure 10.9(g). Moreover, its mean value is not correlated at all with the actual mapping uncertainty.

The last computed estimator is the area occupied by particles in the space [Sta05b], which we compute as the area of the 3D (2D plus heading) ellipsoid resulting from approximating particles with a Gaussian distribution. Apart from this estimator not being mathematically grounded, it does not take into account the map contents either. This implies that, as an example, it can not distinguish between dispersed particles or a few separate modes where particles concentrate. Furthermore, this estimator shows a high variance (see Figure 10.9(h)) while the robot traverses the loop for the second time, whereas the actual uncertainty keeps reduced since the first loop closure.

In these experiments, the time required to compute the EMMI of the RBPF in a 3.2GHz Pentium 4 is about 127ms for a straightforward implementation in C++

and 49ms for an optimized assembly version that exploits the highly parallel structure underlying the EM computation (averaging maps from all the particles is the most time consuming task in EMMI) by the Streaming SIMD Extension 2 (SSE2) instruction set [Inc99]. Our implementation of the joint entropy takes 210ms for the same particle filter, although possibly more efficient algorithms could be implemented for its computation.

10.6.2 EMI-based active exploration

To test the performance of EMI for selecting actions in active robot exploration, we have chosen the information gain-guided exploration framework proposed by Stachniss *et al.* in [Sta05a]. In short, the approach can be described as follows. At each time step we generate a set of potential targets around the robot, each one being a potential action. These points are generated using the most likely map from the RBPF and taking into account the feasibility of the path given the size of the robot and the potential existence of obstacles along that path.

In our implementation we employ a simple path planner that assumes a circular robot and extracts collision-free paths according to a given occupancy grid map. Once we have established those potential paths (actions), we predict the observations along each one of them by means of ray tracing over the grid map. The observations for each path are integrated into a copy of the RBPF, and the change in the uncertainty of the RBPF determines the information gain of each potential action. The robot will take the action with the highest utility value, which is computed by subtracting to the information gain a cost proportional to the length of the path. The navigation towards successive targets is performed by an obstacle avoidance method [Bla06c] that runs on the robot in real-time and concurrently to the update of the RBPF for mapping.

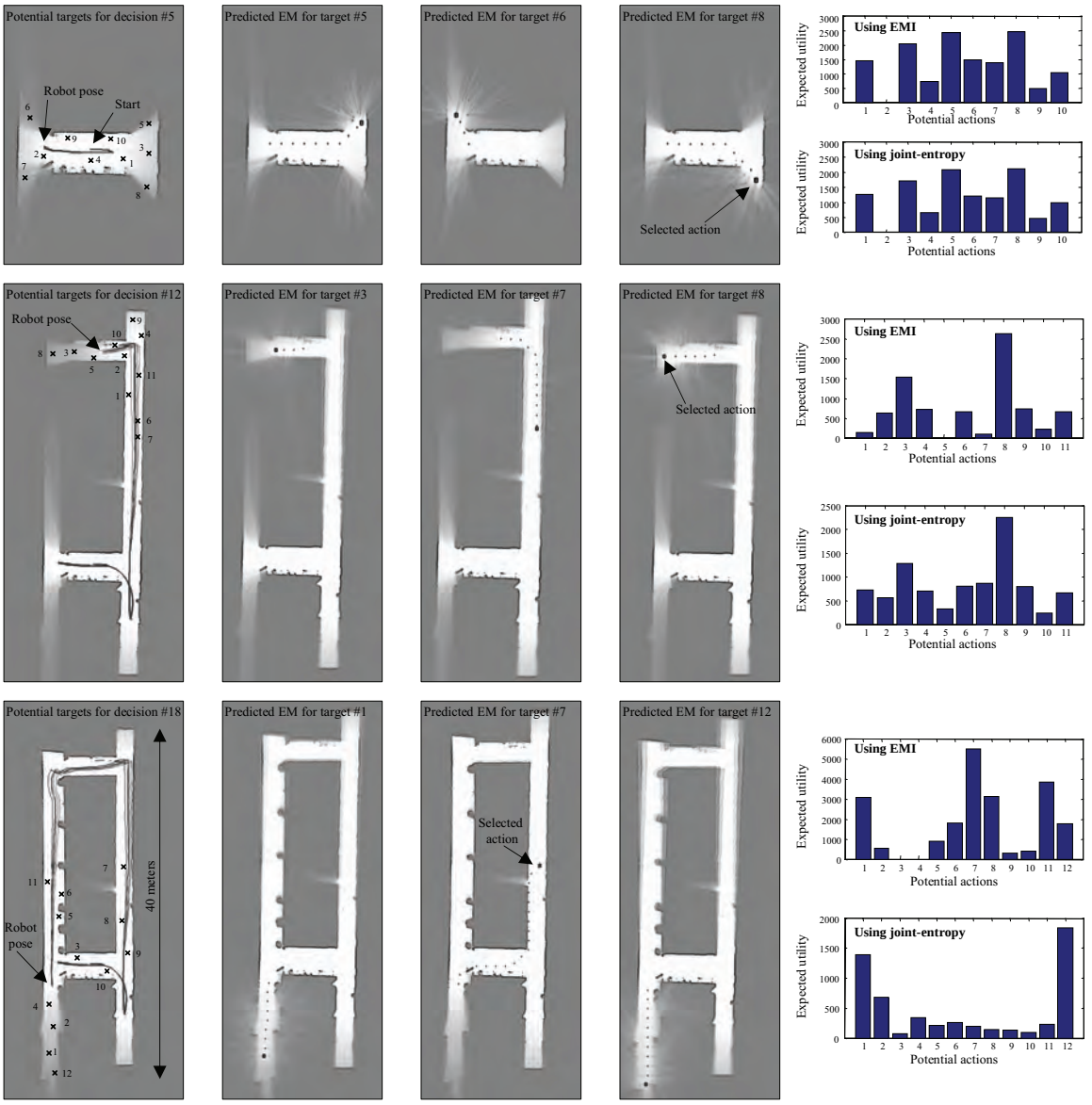


Figure 10.10: Three snapshots of certain moments during an exploration, when the robot has to decide between a set of potential movement actions, marked in the left-hand images. The utility assigned to each one of those actions is plotted both for our EMI measure and the joint entropy in the right-hand graphs, whereas some of the expected maps (EMs) predicted while evaluating the actions are shown in the middle. As discussed in the text, the third decision clearly reflects that actions selected by the EMI are more advantageous.

In our implementation, the gain in information is computed as the increment of the EMI associated to the RBPF, although the decrease of the joint-entropy has been also computed for comparison purposes. Finally, we must remark that for the sake of efficiency, all the computations are performed just for the most likely particle. This approach is acceptable as long as the robot does not transverse too long loops. In those cases, more particles should be considered and the utility values averaged using the particle weights.

In this real experiment, the robot starts in one of the corridors of an environment containing a large loop, as shown in Figure 10.10. For each decision, the robot predicts the EM after the execution of all the potential actions and chooses the action with the highest utility value, as defined above. Some of the so predicted EMs are shown in the middle graphs in Figure 10.10. It can be seen how it is typically predicted that unobserved areas contain free space, which means a future increase in the map information and therefore provides the motivation for exploring them. As the robot gets farther from the starting point, as in the second row in Figure 10.10, the predicted EMs contains areas more and more "blurred", due to the increasing uncertainty in the robot pose. It is noticeably that the utility values obtained from the EMI and the joint entropy for the first and second decisions in Figure 10.10 are very similar to each other. This means that, for actions involving the exploration of new areas, both measures effectively detect which are the most advantageous actions: those leading to the exploration of large unknown regions.

It is interesting to remark here that, as long as the predicted future observations of the robot are random variables, so are the predicted RBPF weights and the corresponding utility values. This helps to explain some apparently strange results in the experiment, such as the assignment of quite different utility values to targets close to each other (e.g. see targets #6 and #7 for the decision #12, or targets #7 and #8

for decision #18). Although we have verified that in spite of not being a problem for exploring relatively small scenarios, it could be solved by predicting future observations a number of times and averaging the resulting utility values.

Consider now the third decision, the bottom row in Figure 10.10. There is a large uncertainty in the robot path after traversing the whole loop, and some of the actions include definitively closing the loop (see target #7, for example), whereas others imply entering new areas to continue the exploration (such as target #12). Interestingly, in this case we obtain quite different values from the EMI and the joint entropy, as can be clearly observed in the bottom-right graphs in Figure 10.10. While the EMI assigns the highest utility to one action that definitively closes the loop (notice the "sharp" EM which is predicted for action #7), the joint entropy assigns the highest values to those paths leading to unknown areas. This is due to the dominance in that measure of the uncertainty in maps, as discussed throughout this chapter, which hides the potential reduction of uncertainty in the robot path derived from the loop closure. Therefore, we can conclude that EMI performs similarly to the joint entropy for exploration, with the additional advantage that it detects much more clearly the possibilities for closing loops, which reduces the overall uncertainty of the SLAM posterior in such a way that the rest of the exploration will be easier than if loops are not closed.

10.6.3 Discussion

In this chapter we have motivated the need for measuring the uncertainty of an RBPF solution to SLAM, which is required for tasks like detecting loop closure or active exploration. We have highlighted the drawbacks of previously employed measures such as the joint path-map entropy. As an alternative, it is proposed to construct the expected map, a new map that condenses all the uncertainty in the RBPF, both for

the map and the robot path. The information of this map is measured by the EMI of the RBPF, which has some advantages in comparison to other measures for choosing actions in active exploration. Moreover, we have also defined the EMMI, which can be applied to the problem of detecting when to stop of actively following an already traversed loop. A priori, it is unknown how much time the robot has to follow a known loop for the incorrect particles to be discarded. To detect when to stop the loop closing behavior, existing approaches either use the area covered by the particles or the effective sample size [Sta04]. None of these methods consider the map contents. We have presented experimental results that demonstrate that EMMI fits better to this purpose than the other methods.

Part III

Large-scale SLAM

CHAPTER 11

OVERVIEW

In this chapter we will discuss the existing works in the fields of metric, topological, and hybrid mapping, highlighting their relation to the new approach presented in this thesis.

Pure metric approaches aim at reconstructing a representation of the environment where the relevant information is the metrical arrangement and characteristics of the map elements. Popular metric representations are landmark maps [Aya89,Dis01,Tar02, Smi90], occupancy grids [Elf89, Gri07a, Mor85], and raw range scans [Gut99, Lu97a] (please refer to [Thr02] for a more detailed classification). Some advantages of metric maps are their direct relation with robotic sensors and their value for some tasks such as motion planning or obstacle avoidance. There exist non-probabilistic approaches for building metric maps [Gon94, Gut99, Lu97a], although most works rely on probabilistic representations of the robot pose and the map and Bayesian filtering is used to

estimate the corresponding probability distributions [Dis01, Smi90]. The hardest problem in those methods is data association (that is, establishing correspondences among observations and the map) [Nei01], a problem that aggravates when the robot closes a loop since the uncertainty in the robot pose and the map increases as the robot explores new areas, i.e. the hardness of finding the correct data association increases with the scale of the maps, and in turn, establishing wrong correspondences severely compromises the consistency of the estimated maps. In fact, during the last years the research on the consistency of EKF-like solutions for SLAM has gained interest, since the approximations introduced by linearizing and assuming Gaussian distributions are known to impose a maximum length for a loop to be correctly closed [Bai06b, Cas04]. The idea of partitioning the environment into a sequence of sub-maps has been proposed as an improvement [Tar02, Paz07], and in fact it became one of the basic motivations of our proposal, named Hybrid Metric Topological (HMT) SLAM.

As already discussed in this thesis, a popular approach to the above mentioned problems of metric mapping is to employ RBPFs. Recall from Chapter 7 that, if we denote the map as m, the robot path as $x_{1:t}$ (or x^t for compactness), and the robot actions and observations as u_t and z_t, respectively, we can observe the following factorization of the SLAM posterior:

$$p(x^t, m|u^t, z^t) = p(x^t|u^t, z^t)p(m|x^t, u^t, z^t) \tag{11.0.1}$$

which is exploited in a RBPF and holds for any form of the probability densities. This factorization shows that, if we estimate the robot path through a separate particle filter, the map m can be estimated from the individual contributions of the observations z_t since they are conditionally independent given x^t. This is supported by the structure of the variables involved in SLAM, as was shown in Figure 7.1, which has been exploited

successfully to build large maps both with occupancy grids [Gri07a] and landmarks (FastSLAM [Mon02a]). However, the number of samples required in these particle filters for closing a loop should be increased with the length of the loop (refer to §8.3), which may eventually turn into a storage capacity limitation since each particle carries a hypothesis of the whole map. Another limitation of RBPF for large-scale global mapping is the loss of particle diversity when closing nested loops. Strategies exist for alleviating these problems [Sta04, Sta05b], but in them the underlying hurdles are just postponed. Another drawback of global mapping with RBPF is that the loss of diversity after closing a long loop typically leads to the total loss of the robot path uncertainty. An enlightening discussion about the problems of standard particle filters for large-scale mapping was recently presented in [MC07].

In contrast to these pure metric approaches, building a topological map is an attractive alternative due to, among other properties, the reduced storage requirements and the good integration with symbolic planning of complex tasks in the AI literature [Cho01b, Dud91, Sav04]. However, pure topological maps are not suitable to solve SLAM. Although Bayesian estimation has been reported recently for these maps in [Ran06], there it is assumed a discrete set of "distinctive" places within the environment which must be correctly detected whenever the robot passes close to them: the only relevant sensory information is that of being close to a distinctive place or not. We believe that a variety of sensors (like laser scanners, or cameras) can provide the robot a more accurate pose estimation through local metric sub-maps than that obtained by a purely topological approach, where most of the sensory data is simply ignored.

So far, it seems an appealing approach that of considering *hybrid maps*[1], where

[1]These maps are sometimes referenced as *hierarchical maps* in the SLAM literature. We use here the alternative term "hybrid" to avoid confusion with pure topological graph representations that involve abstraction processes, which are also usually called "hierarchical" [FM02, Gal04].

topological nodes can contain local metric information [Bos04, Est05, Kou04, Kui04, Mod04, Tom03, Thr96]. In particular, the Atlas framework [Bos04] and the work on hierarchical SLAM by Estrada *et al.* [Est05] contain interesting similarities with HMT-SLAM: both reference uncertainty to local coordinate frames and represent maps as topological graphs with local metric sub-maps. However, in these previous works loop closure has been considered only under the metric point-of-view, i.e. by finding the global metric coordinates transformation compatible with the loop closure. We will show that considering the whole HMT path of the robot leads to important advantages.

In the rest of chapters within this part devoted to large-scale SLAM, we will introduce our hybrid approach to SLAM (in Chapter 12) and other solutions that find a direct application within the framework of HMT-SLAM. In particular, Chapter 13 deals with the partitioning of large metric maps into sub-maps, whereas Chapter 14 addresses the problem of detecting whether a pair of metric sub-maps can be registered to each other.

CHAPTER 12

HYBRID METRIC TOPOLOGICAL SLAM

> "We can only see a short distance ahead, but we can see plenty there that needs to be done."
>
> Computing machinery and intelligence. Mind (1950), vol. 59, pp. 433-460.
>
> Alan Turing

12.1 Introduction

The common formulation of the SLAM problem consists of building some kind of world representation from the sequence of data gathered by the robot, assuming no prior information about the environment and simultaneously localizing the robot using that representation. Different kinds of representations, or *maps*, have been proposed in the robotics and the artificial intelligence literature, ranging from low-level metric

maps, such as landmark maps [Dis01, Smi90] and occupancy grids [Elf89], to topological graphs which contain high-level qualitative information [Cor03, Kui90, Gal05], even multi-hierarchies of successively higher-level abstractions [FM02]. While extant techniques, including those presented in previous chapters, allow building maps of relatively large areas, SLAM remains a largely unsolved problem in relation to high-level representations and long-term operation within large-scale environments.

After an intense research during the last decade, it is clear that the most successful methods for SLAM are those based on probabilistic Bayesian estimation, which can manage well noisy measurements and uncertainty in the robot location, in the map, and, for those based on particle filters, also in data association [Dis01, DW06, Mon02a, Thr02, Thr05]. Therefore, our proposal is grounded on the robustness and accuracy of techniques for metric localization and mapping within small-sized scenarios. In the SLAM literature it has also been shown that large-scale environments can be divided into *areas* of convenient sizes, where these techniques can be applied efficiently to produce consistent *local sub-maps* [Bos04, Est05].

In this and subsequent chapters we will deal with the extension of existing SLAM methods to large-scale scenarios. We will propose that the abovementioned division of space must depend on the nature of sensors, in such a way that each area contains portions of the environment that are very likely to be sensed by the robot simultaneously, whereas parts of different areas will be rarely or never observed at the same time. We will employ for this purpose the method presented in Chapter 13, based on this criterion of simultaneous visibility, or *overlap*. Notice that this kind of "area" does not correspond to symbolic or semantic divisions as could be interpreted by a human [MM07], such as a "corridor" or a "room", but is based solely on the robot's sensory apparatus.

Using this definition of area, we introduce the concept of the hybrid metric-topological

(HMT) path, which comprises the sequence of areas the robot has traversed (topological part) and its pose within each of them (metric part). Therefore, our approach links classical metric SLAM with topological (symbolic) representations typical of artificial intelligence (AI) works.

By considering the posterior belief distribution of the whole HMT path, we can obtain the probability distribution over all the potential *topological structures* of the environment, an issue not addressed before simultaneously to the estimation of the metric poses between, and within, the areas. The resulting probabilistic map, called HMT map, represents the topology of the environment with graphs whose nodes (areas) are annotated with metric sub-maps and whose arcs (connections between areas) are annotated with the coordinate transformations between the corresponding areas. By conditioning the belief distribution of the map to the knowledge of the HMT path, we can represent these relative coordinate transformations in closed form, avoiding by design some problems that appear in global mapping with particle filters [MC07, Sta05b]. The need to avoid absolute coordinates when dealing with large maps has been repeatedly claimed in the literature, due to the difficulty of appropriately representing the uncertainty of poses far away from a global coordinate origin [Bos04, Est05, Fre06, Tar02].

The presented approach, called HMT-SLAM, supports metric sub-maps of either landmarks or occupancy grid-maps. In the context of occupancy grid mapping, it naturally provides a correction of the robot path after closing large loops without rebuilding any global metric map, which has been pointed out sometimes as a weakness of grid-based, large-scale mapping.

Our work is related to some existing methods for hybrid mapping, specially to Hierarchical SLAM [Est05] and to the *Atlas* framework [Bos04]. The fundamental contributions of this new proposal in the context of these and other previous works

are:

- The introduction of probability distributions over both the metric *and* the topological part of the robot path. Previous works have considered either the metric [Gri07a, Mur99, Mon02a] or topological [Ran06] paths separately. Apart from the mathematical consistency of a unified Bayesian approach, this formulation supports multiple topological hypotheses, and can be factored in such a way that allows the uncertainty of large maps to be maintained accurately.

- A statistically grounded principle for the separation of the map into sub-maps. In [Est05], new sub-maps are started if the previous ones reach a given number of landmarks, while in [Bos04] this is performed whenever a measure of the localization performance degrades. We propose instead to generate sub-maps that minimize a given measure[1] of covisibility or overlapping between groups of observations, which allows us to set a grounded statistical model for HMT-SLAM as a Bayesian inference problem. Also, this provides the robot with topological structures that do not depend on external engineered knowledge, but on its own sensory system.

- In contrast to previous works (such as [Bos04]), in HMT-SLAM all the hybrid map hypotheses are treated equally, associating different metric sub-maps to a given area if there exist multiple hypotheses about the topological path followed by the robot. This implies that the metric pose of the robot may be distributed around multiple modes even for particles with the same topological position. Our approach does not impose limits to the variety of the inferred distributions, whose complexity will uniquely be determined by the ambiguity of the environment and the uncertainty introduced by closing long loops.

[1] Ideally, the cross-covariances in the case of landmark maps.

Finally, we should also highlight that the meaning of the topological part of an HMT map strongly differs from that considered in many other works. In the literature we can find works that consider distinctive places as nodes [Cho01b,Kui90,Kui04], while others cut the map into disjoint areas [Est05, Thr96]. Instead, our approach is closer to those where topological nodes are the result of *abstracting robot observations* gathered at a given area [Bla06b, Ziv05]. Therefore, the size of the areas is automatically determined by the nature of sensors, more concretely, by the overlap between observations [Bla06b]. As an example, exploring a single room with a narrow field-of-view camera may result in many areas, whereas a wide-angle laser scanner might probably lead to only one. This topic will be discussed in more detail in Chapter 13.

This chapter is structured as follows. Firstly, we provide in §12.2 the probabilistic foundations for our approach, while some of its key elements are discussed in depth in §12.3. A practical system that implements these ideas is presented in §12.4, and experimental results with real robots in large-scale scenarios, as well as comparisons to other approaches, are discussed in §12.5.

Table 12.1: Summary of the notation employed in this chapter

Symbol	Meaning
m	The HMT map (an annotated graph).
${}^{a}\mathcal{M}$	The local metric map for the area a.
${}^{b}_{a}\Delta$	The coordinate origin of area b relative to that of area a.
s_t	The robot HMT pose at time step t.
u_t, o_t	The robot actions and hybrid observations at time step t.
s^t, u^t, o^t	The sequences of robot poses, actions, and observations for time steps 1 to t.
${}^{i}s_{t'}, {}^{i}u_{t'}, {}^{i}o_{t'}$	A convenient way of referencing the robot poses, actions, and observations grouped into the area i such as the first elements are given for $t' = 0$.
${}^{i}s^t, {}^{i}u^t, {}^{i}o^t$	The sequences of all the corresponding variables up to time step t.
ψ_t, z_t	The area-dependant and metric observations, respectively.
γ_t, x_t	The topological and metric parts of s_t at time step t, respectively.
γ^t	The topological path of the robot up to time step t.
Υ_t	The set of all known areas at time step t.
$s_t^{[k]}$	The k'th particle at time step t for the robot HMT pose.
$\omega_t^{[k]}$	Importance weight of the k'th particle at time step t.

12.2 Probabilistic foundations of HMT-SLAM

The present proposal for HMT-SLAM is grounded on the sparsity of the relations existing among robot observations in a large environment: by our definition of *area*, observations within a given area will be highly related to one another, while observations belonging to different areas will not. This fact has already been employed in a number of works to ease the construction of landmark maps [Fre05, Liu03].

We start by defining an HMT map m of the environment as an annotated graph[2] that comprises the 2-tuple:

[2]By "annotated graph" we mean a graph whose nodes and arcs can hold some non-topological information. This is an informal version of the term "typed attributed graph" in the graph transformation literature [Ehr04].

$$m = \langle \{ {}^{i}\mathcal{M} \}_{i \in \Upsilon_t}, \{ {}^{b}_{a}\Delta \}_{a,b \in \Upsilon_t} \rangle \tag{12.2.1}$$

Here the ${}^{i}\mathcal{M}$ are metric sub-maps for each area $i \in \Upsilon_t$, where Υ_t stands for the set of known areas at time step t. On the space of these hybrid maps, we will define beliefs as probability distributions. The local maps and the coordinate transformations ${}^{b}_{a}\Delta$ between the adjacent areas a and b are the annotations of the nodes and the arcs in the graph, respectively. Although this is the only information relevant for the following discussion, it is worth mentioning that the graph could also maintain data about the kind each area is, the navigability between areas, or any other high-level knowledge useful in graph-based planning or symbolic reasoning [Gal04].

We will consider conditional probability distributions over m given the information gathered by the robot up to some time step, where each particle may contain in turn different belief distributions for both the metrical sub-maps and the coordinate transformations, and even a different number of nodes and arcs. Accordingly, the robot is provided with a hybrid discrete-continuous description of its position within the map. Let $s_t = \langle \gamma_t, x_t \rangle$ denote the robot HMT pose at time step t, where the discrete variable γ_t indexes the current area (node in the graph) while the continuous variable x_t represents the metric pose relative to the local coordinate frame of that area, as represented schematically in Figure 12.1.

We can now state the problem of HMT-SLAM as the estimation of the joint posterior of the robot HMT path s^t and the map m given the sequences of robot actions u^t and observations o^t up to time step t, that is:

$$p(s^t, m | u^t, o^t) \tag{12.2.2}$$

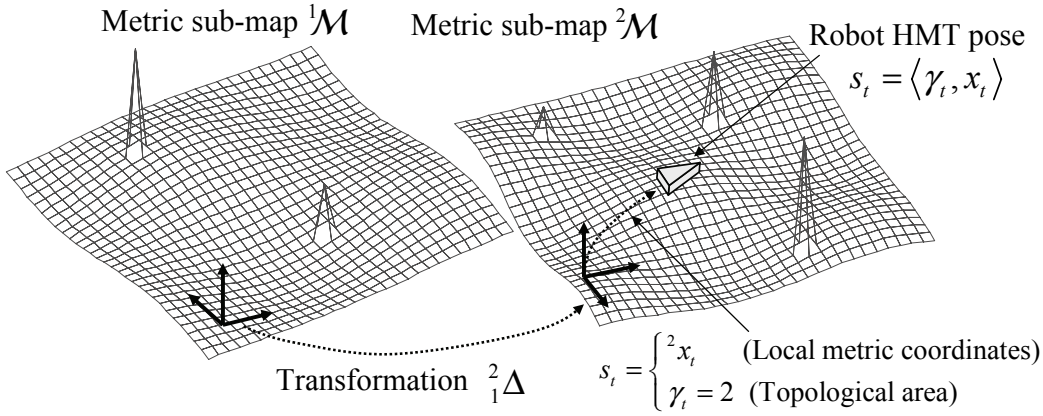


Figure 12.1: Each metric sub-map ${}^i\mathcal{M}$ has its own local coordinate frame. The robot HMT pose is thus a discrete-continuous variable which states both the current area (the node in the topological part of the map) and the metric pose within the corresponding local frame.

As usual, we denote sequences of variables over time with superscripts, i.e. $u^t = \{u_1, ..., u_t\}$. We also define o_t as containing the hybrid pair $\langle \psi_t, z_t \rangle$ with the purpose of conveniently setting metric observations z_t (such as range scans or landmarks extracted from images) apart from topologically dependant observations ψ_t (such as the recognition of a particular kind of area). This division renders especially useful when facing the problem of global localization.

In this thesis, HMT-SLAM is addressed under the point-of-view of Bayesian sequential estimation, using the graphical model proposed in Figure 12.2. An alternative approach is to estimate the whole HMT robot path at each time step, an idea that needs to be developed in future works.

For clarity, the first time step at each segment of the path that form one area has been denoted as 0, and we add the current area as a left superscript to the name of variables, e.g. replacing s_t by ${}^{\gamma_t}s_{t'}$ where the primed t' reflects that this is a different time index than the "global" t.

We can observe the following interesting probabilistic properties in the structure of HMT-SLAM under the proposed sequential estimation model.

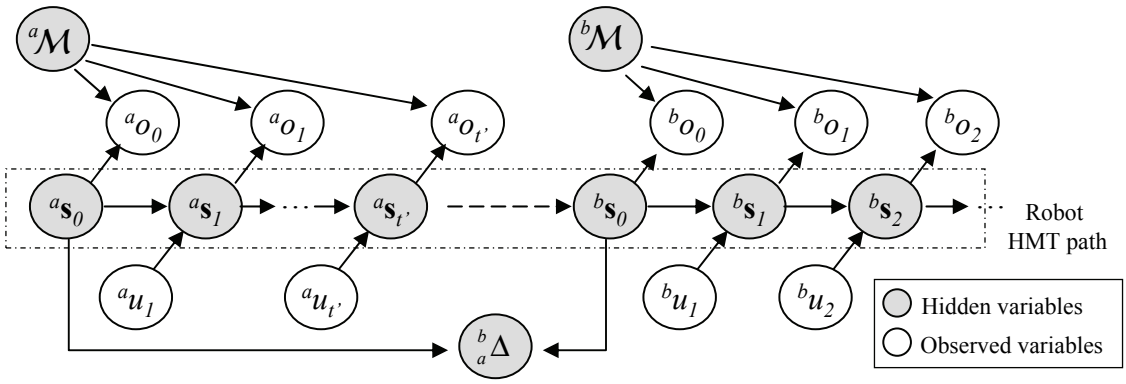


Figure 12.2: The graphical model for HMT-SLAM. Here segments of the robot path are conditionally independent given the starting pose at each segment. The relative pose between areas is represented by ${}^{b}_{a}\Delta$, while the term ${}^{i}\mathcal{M}$ stands for the metric sub-map of area i.

Proposition 1. Given the distribution of the robot HMT path s^t, and due to the conditional independence between clusters of observations that belong to different local sub-maps, the joint posterior of all the sub-maps $i \in \Upsilon_t$ can be factored as:

$$p(\{{}^{i}\mathcal{M}\}_{i\in\Upsilon_t}|s^t, u^t, o^t) = \prod_{i\in\Upsilon_t} p({}^{i}\mathcal{M}|{}^{i}s^t, {}^{i}o^t) \tag{12.2.3}$$

This property holds in general, regardless of the choice of the metric map representation.

This conditional independence between the sub-maps given the robot path can be clearly seen in Figure 12.2, where the robot path d-separates (see §2.10) all the possible pairs of sub-maps.

Proposition 2. The metric sub-maps ${}^{i}\mathcal{M}$ and the coordinate transformations ${}^{b}_{a}\Delta$ are conditionally independent given the robot HMT path.

This second property can be verified in Figure 12.2 by noticing that the robot path also d-separates the set of metric sub-maps and the set of arc transformations ${}^{b}_{a}\Delta$.

Proposition 3. The posterior distribution of the robot HMT path s^t can be factored into the product of the posterior of its segments through the different areas i:

$$p(s^t|u^t, o^t) = \prod_{i \in \Upsilon_t} p({}^i s^{t'}|{}^i u^{t'}, {}^i o^{t'}) \tag{12.2.4}$$

The reason of this third property is that consecutive robot poses grouped into different areas, like ${}^a s_{t'}$ and ${}^b s_0$ in Figure 12.2, become independent variables in our model. This may seem surprising at first glance, since these poses will often represent close, consecutive positions along the robot metric path, and they are actually related by a robot action (e.g. odometry) and observations of roughly the same place. Thus, this assumption of independence between *pure metric poses* might seem to certainly lead to a significant loss of information. The key point here is that we assume their independence as *hybrid robot poses* (metric plus topological), where the metric coordinates are referenced to different local frames: the information from the last robot action (${}^a u_{t'}$ in Figure 12.2) is not lost, but incorporated into the corresponding ${}^{b}_{a}\Delta$ variable.

Nevertheless, it must be noticed that the cross-covariance between robot poses at different areas is definitively lost in HMT-SLAM. In practice, partitioning the map will rarely generate strictly independent observations between local maps, which renders our approach as an approximate solution to SLAM. The loss of information, however, can be minimized as much as desired by using grounded methods in the process of deciding when to start a new area [Bla06b, Ziv05]. As a limit situation, if we desire no loss of information at all, HMT-SLAM degenerates into the common global metric SLAM.

For simplicity, we also assume here that the methods for defining an area give us roughly the same results independently of the concrete robot path to reach that area. This is required for identifying two different areas as the same physical placement, a requirement for closing loops at the topological level. Although this may lead to sub-optimal partitions, our partitioning method will always generate the best ones in terms of the maximum independence between observations. Our experimental results show that, under the common assumption in SLAM of a quasistatic world, the present approach to HMT-SLAM is appropriate for practical situations.

Based on the above probabilistic properties of HMT-SLAM, we propose the following solution to the problem of estimating the posterior of Eq. (12.2.2). We start by using the Rao-Blackwellization approach, described in Chapter 7, to first factor the joint posterior into one component for the robot HMT path s^t and another one for the map m:

$$p(s^t, m|u^t, o^t) = p(s^t|u^t, o^t)p(m|s^t, u^t, o^t) \tag{12.2.5}$$

In this way we reduce the dimensionality of the joint path-map space to that of the robot HMT path only (s^t), which can then be estimated using a particle filter. Then, for each particle, it is computed the analytical conditioned distribution of the map (m) associated to the corresponding path hypothesis. Writing down the analytical part of Eq. (12.2.5), and given the conditional independence between the elements of m (proposition 2 above), we have:

$$\begin{aligned} p(m|s^t, u^t, o^t) &= p(\{{}^i\mathcal{M}\}_{i\in\Upsilon_t}, \{{}^b_a\Delta\}_{a,b\in\Upsilon_t}|s^t, u^t, o^t) \\ &= p(\{{}^i\mathcal{M}\}_{i\in\Upsilon_t}|s^t, o^t)p(\{{}^b_a\Delta\}_{a,b\in\Upsilon_t}|s^t, u^t, o^t) \end{aligned} \tag{12.2.6}$$

As shown in proposition 1, the first distribution in this product can be factored and each sub-map computed using existing closed-form equations for landmark maps [Mon02a] and occupancy grids [Mor88, Thr03]. The second element in Eq. (12.2.6) is the joint posterior of the variables ${}^{b}_{a}\Delta$. As typically done in the large-scale SLAM literature, we use Gaussians to model these relative poses, that is:

$$ {}^{b}_{a}\Delta \sim \mathcal{N}({}^{b}_{a}\bar{\Delta}, {}^{b}_{a}\Sigma) \tag{12.2.7} $$

We can now perform a Bayesian fusion in closed-form for each arc ${}^{b}_{a}\Delta$ in the HMT map simply by [3] :

$$ p({}^{b}_{a}\Delta | s^t, u^t, o^t) \propto \prod_{j=1}^{L} \mathcal{N}({}^{b}_{a}\bar{\Delta}'_j, {}^{b}_{a}\Sigma'_j) \tag{12.2.8} $$

where L is the number of times the robot has moved between the areas a and b. The parameters of the j'th Gaussian in Eq. (12.2.8) can be obtained from the metric path of the robot (when the robot explores new areas), or from map alignment procedures (when considering the hypothesis of a topological loop closure). This means that each time the robot moves between a given pair of areas, the estimation of their relative pose is refined and the uncertainty is thus reduced. Note as well that each variable ${}^{b}_{a}\Delta$ can be estimated independently as a consequence of proposition 3, which is consistent with the construction of a map of relative poses. Therefore,

$$ p(\{{}^{b}_{a}\Delta\} | s^t, u^t, o^t) = \prod_{a,b} p({}^{b}_{a}\Delta | s^t, u^t, o^t) \tag{12.2.9} $$

[3] This simple equation can be derived by considering the estimation of each arc value ${}^{b}_{a}\Delta$ as a Kalman filter (KF) where each loop closure in HMT-SLAM becomes an observation for this "auxiliary KF". Operating on the usual KF equations one arrives at Eq. (12.2.8).

Now we address the non-analytical estimation of the robot path, the first term in Eq. (12.2.5). We estimate this posterior sequentially by Bayesian filtering:

$$p(s^t|u^t, o^t) \propto p(o_t|s^t, u^t, o^{t-1})p(s^t|u^t, o^{t-1}) \tag{12.2.10}$$

Like in [Ran06], a particle filter is employed as a convenient representation of the topological part of the robot path (γ^t). Therefore, if we assume a set of P weighted particles distributed approximately according to the posterior for time step $t-1$:

$$\{s^{t-1,[k]}\}_{k=1..P} \sim p(s^{t-1}|u^{t-1}, o^{t-1}) \tag{12.2.11}$$

we can sequentially generate the particles for the next time step t by drawing samples from a given proposal distribution $q(s_t|s^{t-1}, o^t, u^t)$ and updating the importance weights $\omega_t^{[k]}$ accordingly. We consider here the optimal proposal for each particle to be $p(s_t|s^{t-1,[k]}, o^t, u^t)$, which has been demonstrated to minimize the variance of the weights as discussed in chapters 4 and 8. Having an exact expression for this proposal means that the generated particles will be distributed according to the true posterior. As shown in the derivation in Eq. (12.2.12) below, we can put the optimal proposal for our particle filter in a form that allows us to sample a new particle $s_t^{[k]}$ in two steps: firstly, to draw the topological position $\gamma_t^{[k]}$ using a topological transition model (discussed in a later section), and then, to draw the metric pose $x_t^{[k]}$ from the conditional distribution of the obtained topological position.

$$
\begin{aligned}
s_t^{[k]} &= \left\langle \gamma_t^{[k]}, x_t^{[k]} \right\rangle \sim q\left(s_t | s^{t-1,[k]}, u^t, o^t\right) \\
&= p\left(s_t | s^{t-1,[k]}, u^t, o^t\right) \overset{\text{Bayes}}{\propto} p\left(s_t | s^{t-1,[k]}, u^t, o^{t-1}\right) p\left(o_t | s_t, s^{t-1,[k]}, u^t, o^{t-1}\right) \\
&= p\left(\gamma_t, x_t | s^{t-1,[k]}, u^t, o^{t-1}\right) \overbrace{p\left(z_t | s_t, s^{t-1,[k]}, u^t, o^{t-1}\right) p\left(\psi_t | \gamma_t, s^{t-1,[k]}, u^t, o^{t-1}\right)}^{\text{Independence between } z_t \text{ and } \psi_t} \\
&= \overbrace{P\left(\gamma_t | s^{t-1,[k]}, u^t, o^{t-1}\right) p\left(x_t | \gamma_t, s^{t-1,[k]}, u^t, o^{t-1}\right)}^{\text{Definition of conditional probability}} \cdot \\
&\quad p\left(z_t | s_t, s^{t-1,[k]}, u^t, o^{t-1}\right) p\left(\psi_t | \gamma_t, s^{t-1,[k]}, u^t, o^{t-1}\right)
\end{aligned}
$$

$$
\longrightarrow \quad \begin{array}{ll} 1^{\text{st}} \text{ step:} & \gamma_t^{[k]} \sim P\left(\gamma_t | s^{t-1,[k]}, u^t, o^{t-1}\right) p\left(\psi_t | \gamma_t, s^{t-1,[k]}, u^t, o^{t-1}\right) \\ 2^{\text{nd}} \text{ step:} & x_t^{[k]} \sim p\left(x_t | \gamma_t^{[k]}, s^{t-1,[k]}, u^t, o^{t-1}\right) p\left(z_t | x_t, \gamma_t^{[k]}, s^{t-1,[k]}, u^t, o^{t-1}\right) \end{array} \tag{12.2.12}
$$

This procedure is supported by our knowledge of the conditioned density of the metric pose given the topological position. The metric sample can be obtained from exact equations for landmark maps [Mon03b] or from approximations for occupancy grids [Gri07a]. This process is repeated for each time step, performing resampling [Rub88] whenever the effective sample size [Liu96] of the RBPF falls below a given threshold, e.g. the 50%. Drawing hypotheses for the topological position γ_t is arguably the most complex step in HMT-SLAM. Later on we discuss an implementation aiming at real-time execution which has given good results.

Observe that each hypothesis of the topological path γ^t implies a different topological structure of the environment (and thus, a different hypothesis for m) by means of rearranging the sequence of areas traversed by the robot [Ran06]. Since each particle may maintain a different HMT hypothesis, we have a probability distribution over the possible topologies of the map, as illustrated with a few examples in Figure 12.3.

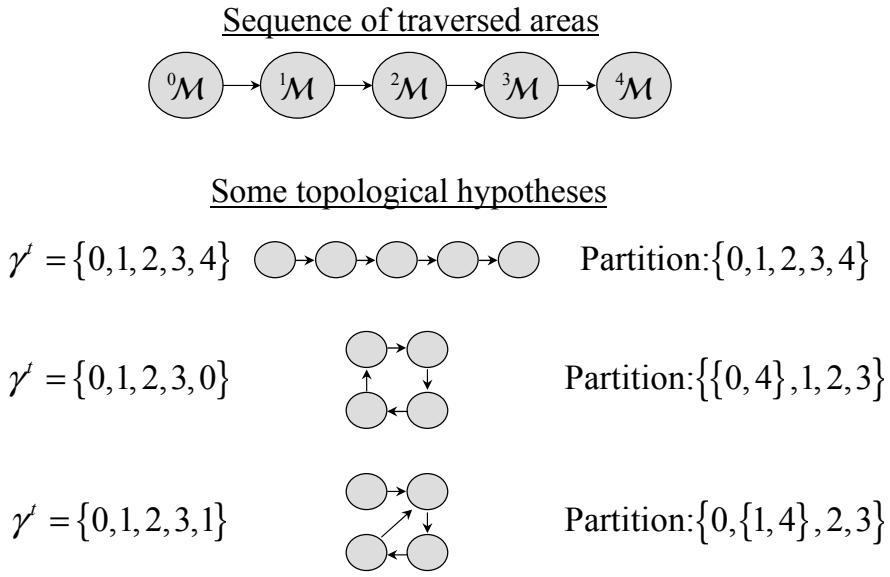


Figure 12.3: Our approach takes the sequence of areas traversed by the robot (on the top), and estimates the topological structure of the environment by considering some of the potential rearrangements of the sequence. The bottom graphs show some examples of partitions (rearrangements), the associated topological paths γ^t, and the resulting map topologies.

12.3 Relevant elements in HMT-SLAM

After discussing the theoretical foundations of HMT-SLAM, in this section we deal with some practical issues such as the topological transition model or how to obtain a global map from an HMT map.

12.3.1 The transition model of the topological position

A fundamental part of HMT-SLAM is the estimation of the topological path of the robot. In this thesis this problem has been addressed sequentially via particle filtering and considering the optimal proposal distribution [Dou00b] for generating new hypotheses. As shown in Eq. (12.2.12), this leads to drawing samples $\gamma_t^{[k]}$ from the product of two terms: the transition model of the topological position $P\left(\gamma_t | s^{t-1,[k]}, u^t, o^{t-1}\right)$, denoted in the following as $P\left(\gamma_t | d^{[k]}\right)$ for clarity, and the appearance observation model

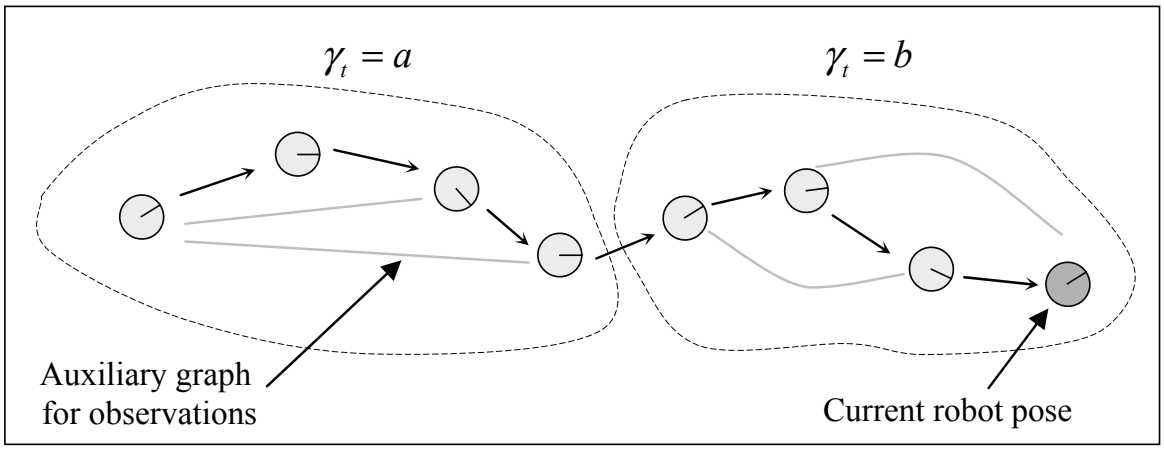


Figure 12.4: An auxiliary graph of robot observations can be used to detect when the robot enters into a new area through graph bisection techniques.

$p\left(\psi_t|\gamma_t, s^{t-1,[k]}, u^t, o^{t-1}\right)$. Some possible implementations of the latter have been reported by other authors [Cum07], although that part has not been integrated in the present work yet.

Due to its discrete nature, the topological transition model assigns a probability to a reduced number of potential events:

- To stay at the same topological area (simply $\gamma_t^{[k]} = \gamma_{t-1}^{[k]}$).
- To enter into an unexplored area, $\gamma_t^{[k]} \notin \Upsilon_{t-1}$.
- To close a topological loop, that is, $\gamma_t^{[k]} \in \Upsilon_{t-1}$ with $\gamma_t^{[k]} \neq \gamma_{t-1}^{[k]}$.

The topological position of the robot γ_t is therefore a piecewise constant sequence over time t. For example, consider the robot path shown in Figure 12.4, where the robot topological pose is a or b in the two separate parts of the path.

Therefore, it is essential to recognize the *transitions* of the robot from one area to the another, what can be achieved by existing methods for partitioning a sequence of robot observations that minimize some measure of similarity or overlapping between them (the exact measure to minimize may depend on the actual sensors and

map representations [Bla06b, Ziv05]). In short, this method builds an auxiliary graph whose nodes are the robot poses where observations were taken, and whose undirected weighted arcs keep the measure of similarity between the observations, as represented in Figure 12.4. Then, the minimum normalized cut of that graph can be computed by means of spectral bisection [Shi00]. Lower cut values mean weaker relation between the observations in the two subgraphs. A more detailed exposition of this process will be given in Chapter 13. Thus, the robot has moved to a different area if the resulting cut is below a certain limit that settles the maximum allowed dependence between adjacent sub-maps. If such a partitioning happens, the immediate past of the robot path is updated to the new topological position. Note that this process needs to be done only once for groups of particles that share the same topological path.

Going back to the topological transition model, we can now specify that when the above partitioning method does not find a sufficiently independent partition of the auxiliary graph (that is, the robot has not entered a new area), we have $P\left(\gamma_t|d^{[k]}\right) = 1$ for $\gamma_t = \gamma_{t-1}^{[k]}$, and $P\left(\gamma_t|d^{[k]}\right) = 0$ otherwise. In contrast, when a new area is entered, the transition model considers the possibility of the robot having closed a loop by means of $P\left(\gamma|d^{[k]}\right) \propto F(\gamma)$, and the possibility of having entered into an unexplored area by $F(\gamma') = \eta$, where η is a constant (all the probabilities are scaled such as they sum up to unity). The function $F(\gamma)$ provides a measure of how likely it is to have arrived at the previously known area γ through a loop closure. This measure should incorporate the metric information of the arcs along the topological path between the two nodes (refer to Figure 12.5), although it could also rely on some high-level topological reasoning [Dud91, Rem04]. An interesting choice for this function is the likelihood of the last observations in relation to the metric sub-map of the candidate area γ.

Notice that we will need the exact robot pose *after* closing the loop to compute

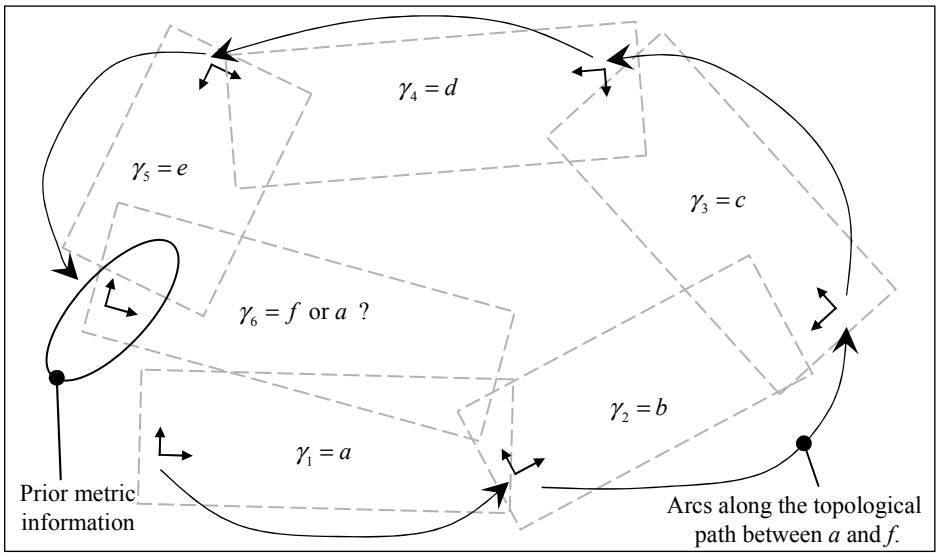


Figure 12.5: Is the robot after the loop defined by the solid arcs at area a or at f? The topological transition model assigns a probability to each of these possibilities by accounting for the robot observations and the prior metric information (and uncertainty) contained in the arcs of the HMT map.

this measure, which might seem problematic since the aim of this computation was just determining potential loop closures! One of the possible ways to overcome this is to follow the grid matching method described in Chapter 14, suitable to the case of metric maps given as grid maps. Only some of the areas γ will be determined as potential loop closures by the grid matching procedure, and in those cases an estimate of the robot pose *after* the loop closure will be obtained which can then be employed to evaluate the above-mentioned observation likelihoods needed to compute $F(\gamma)$. It should be remarked that the grid-matching must not be evaluated against all the existing areas in the HMT map, but only to those with a reasonable probability of overlapping the current topological area. This can be easily determined by taking into account the extension of grid maps and their relative positions ${}^{b}_{a}\Delta$, whose pdf computation is described in §12.3.3. Notice that as the number of areas in an HMT map increases, so does the time complexity of determining potential topological loop closures. This is a

complex issue which requires further research. One of the enhancements that needs to be explored is the usage of additional hierarchy levels of areas [Gal04]: for instance, a second level of abstraction might collect all the areas within a "building" in such a way that when a robot searches for loop closure candidates it should automatically discard all those out of the current "building".

To sum up, the procedure described in this section assures that loop-closure candidates are within the scope of the metric uncertainty of the HMT map and they also provide a good prediction of the last observations.

12.3.2 Probability distribution over topologies

Although it is not needed by HMT-SLAM, it is sometimes desirable to compute the discrete probability mass function (PMF) of the topological path γ^t, for example for visualizing the topological structure hypotheses considered by the filter at a certain instant of time. This operation can be achieved by marginalization, which for our particle filter simply becomes:

$$P(\gamma^t = \tilde{\gamma}^t | u^t, o^t) = \sum_{k \in \Omega} \omega_t^{[k]} \quad , \Omega = \{k : \gamma^{t,[k]} = \tilde{\gamma}^t\} \tag{12.3.1}$$

for each desired value of $\tilde{\gamma}^t$. Some examples of these distributions will be shown later on in Figure 12.9.

12.3.3 Recovering global metric maps from HMT maps

During the normal operation of the robot using HMT-SLAM it will usually be enough to reference the robot coordinates to the coordinate frame of the current area. However, we could desire sometimes to compute absolute coordinates relative to some arbitrary

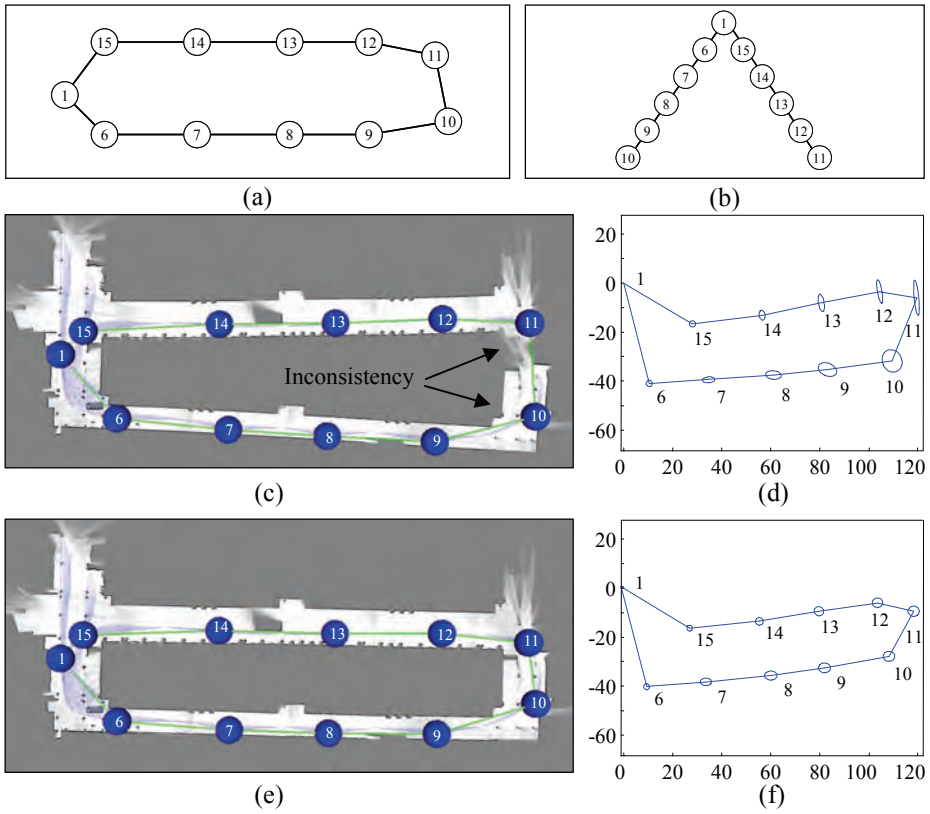


Figure 12.6: (a) An example of an HMT map after closing a loop. (b) A tree representation for finding the shortest path from node 1 to any other node. (c)–(d) The global map and the uncertainty for each area obtained by using the shortest-path method for computing global coordinates. (e)–(f) The same map obtained by using a globally consistent method. The ellipses represent 95% confidence intervals magnified by factor 5 for clarity.

reference, for example for constructing a global map for debugging or simple visualization. We describe next two possible methods for computing the absolute coordinates of all the areas of a map taking one of them as reference.

The first approach, proposed by Bosse *et al.* in [Bos03] and adopted in an early version of our approach [Bla07b], consists of applying the Dijkstra algorithm for finding the shortest topological path from the reference area to the rest. In this way, the topological part of an HMT map is transformed into a tree that encodes all these "optimal" paths, as shown with an example in Figure 12.6(a)–(b). If we denote the reference area as a, and the area we are interested in as b, we will find the corresponding

coordinate transformation ${}^{b}_{a}\Delta$ in the HMT map only if a and b are adjacent areas. Otherwise, this pose can be computed by sequentially composing pose changes along the shortest topological path $\{a, 1, 2, 3, ..., n, b\}$, that is:

$$ {}^{b}_{a}\Delta = {}^{1}_{a}\Delta \oplus {}^{2}_{1}\Delta \oplus ... \oplus {}^{b}_{n}\Delta \tag{12.3.2} $$

where $\oplus$ is the metric pose composition operator [Smi88]. Given the conditional independence between all the relative poses along the topological path, a Monte Carlo simulation can be employed to generate samples of ${}^{b}_{a}\Delta$ from independent samples of all the elements in Eq. (12.3.2). The problem with this approach is that the existence of loops leads to inconsistent global coordinates, since the global coordinates of two adjacent areas may be computed using completely separate topological paths. To illustrate this issue, please refer to Figure 12.6(c)–(d), which represent the global coordinates and the associated uncertainties for each area of a real HMT map. The information in the arc between nodes 10 and 11 has not been considered by the shortest-path approach, hence we find an inconsistency in the resulting map between these two areas.

To solve this, we propose the alternative approach of applying methods for globally consistent pose estimation that has been reported in the past for networks of laser scans [Lu97a]. In this kind of methods, we iteratively compute the approximate optimal poses of all the nodes by minimizing the linearized version of a cost function which includes all the constraints between adjacent areas. This algorithm typically converges in a few iterations, and the so obtained global maps are free of inconsistencies, as can be observed for our example in Figure 12.6(e)–(f). This is the method employed for the rest of global maps shown in the figures.

12.3.4 Long-Term Operation

It is worth to highlight a crucial property of HMT-SLAM that makes it specially suitable for the long-term operation of a mobile robot. The statement of HMT-SLAM as the estimation of the density in Eq. (12.2.2) includes the robot HMT path s^t, whose dimensionality always increases over time. In this sense, it may seem that our approach suffers from the same problem that global mapping with RBPFs, i.e. performing estimation into a state-space of unbounded dimensionality. However, hypotheses for the whole topological path γ^t can be forgotten until the point where the differences between the particles begin. That is, if at a given instant of time all the particles agree about the current topological structure, there is no need to maintain in memory several map topologies with their corresponding metric local maps. Unlike the case of purely metric SLAM, this does not mean a loss of the estimated uncertainty along the robot path, since in HMT-SLAM this spatial uncertainty is maintained in a parameterized form by the conditional distributions for ${}^b_a\Delta$ (refer to Eq. (12.2.8)).

12.4 Implementation framework

In §§12.2–12.3 we have introduced the theoretic formalization of HMT-SLAM. In the following we present a practical framework which implements those ideas. A relevant issue here is that a mobile robot may demand accurate metric localization in hard real-time (e.g. for navigation or manipulation purposes), while maintaining the consistency of the topological map (i.e. solving loop closures) can be performed in the "background" since reasonable delays are acceptable.

An overview of the proposed framework is presented in Figure 12.7, where we can observe its layered structure. Metric local SLAM is performed at the low level, while

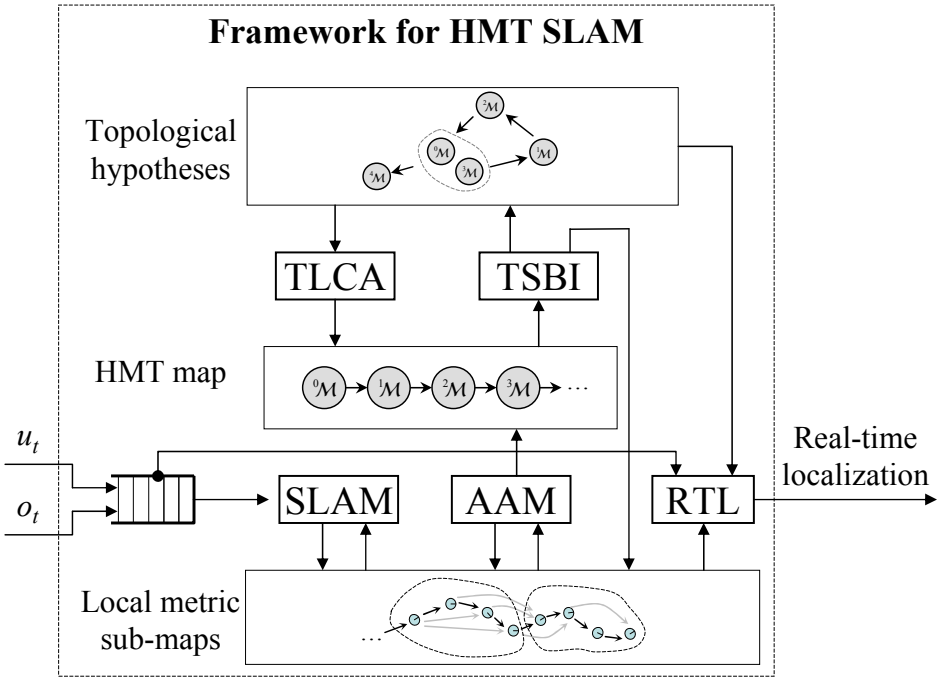


Figure 12.7: Overview of the implementation proposed as a practical solution to HMT-SLAM. Here local metric SLAM is solved efficiently in real-time, while the topological structure of the map is estimated concurrently in a delayed fashion. In this way we prioritize the update of the metric position of the robot within the current area. Please refer to the text for further details.

topological representations are managed at the upper levels. Symbolic reasoning or task planning would fit in additional layers above these. The inputs to the system are actions u_t (typically produced by a planning level), and observations o_t (acquired by the robot), which are kept in a time-stamp-ordered queue until they can be processed. Within the system, there are a number of processes running concurrently which interact by means of read and write operations on the data held in the three levels represented in Figure 12.7: the local metric map of the current area, the sequence of traversed areas, and the space of topological path hypotheses. It must be remarked the parallel nature of the system, that is, the processes do not need to run in a predefined, sequential order. Next we describe these processes and their relations with the theory presented previously.

Metric SLAM: It handles the robot localization and mapping within the metric sub-map for the current area by processing actions and observations. Conceptually, this process is involved in estimating the metric part of Eq. (12.2.12). For occupancy grids, RBPFs have a complexity linear with the number of particles. Therefore, for a static number of particles we have a constant computation time independently of the size of the local area, which is a requisite if we desire a hard real-time estimation. In practice, we would need a variable number of particles depending on the number of topological path hypotheses considered at each instant of time, which may increase after closing long loops. If we want this metric SLAM to have a bounded time complexity even in those cases we could impose a maximum number of particles in the filter, at the cost of disregarding the most unlikely topological hypotheses in the TSBI process (described below).

Area Abstraction Mechanism (AAM): Grounded methods are applied here to detect clusters of (approximately) independent observations in the sequence of observations gathered by the robot [Bla06b], as briefly described in §12.3.1 and addressed in depth in Chapter 13.

Topological Space Bayesian Inference (TSBI): The topological transition model described in §12.3.1 is applied here whenever the AAM detects that the robot has moved into a new area. In our current implementation, the local metric-maps for the particles are not updated until TSBI gives us the hypotheses for the new topological position of the robot. Once the topological transition model has been evaluated, all the particles are updated to account for the changes, which includes changing the coordinate references, removing part of the current metric sub-map in the case of entering into a new area, and in the case of a loop closure, loading the contents of the known area into the sub-map. Regarding the computational complexity of the TSBI, a straight-forward implementation exhibits quadratic complexity in the number

of known areas $|\Upsilon_t|$. This complexity is imposed by the Dijkstra algorithm used in the computation of the a priori metric information (see 12.3.1).

Topological Loop Closure Acceptance (TLCA): As discussed above, it is unpractical to keep the whole history of the hybrid path for long-term operation. The TLCA process is in charge of accepting part of the topological structure as definitively correct. To avoid the risk of losing the valid (unknown) hypothesis in this process, it only operates when the system contains highly dominant (or unique) hypotheses for the robot topological path γ^t.

Real Time Localization (RTL): This process guarantees an estimation of the robot position in a timely fashion, if that is really needed. Notice that this stage is optional and it not specificaly linked to our HMT-SLAM proposal: this stage might be perfectly applicable to any other SLAM method as well. Here, if the input queue of actions and observations is empty, the best estimation of the HMT pose s_t has been already updated by our HMT-SLAM algorithm. On the contrary, when there are pending actions in the queue, the RTL computes the next prior distribution, e.g. $p(s_{t+1}|s_t, u_{t+1})$, as a more updated estimation of the actual robot pose. We can easily compute this prior for the robot metric pose in real-time, for example, by accumulating odometry readings. The obtained pose estimations will not be the optimal ones, but as long as the metric SLAM process updates its estimation in a timely fashion, the estimate from RTL will be simultaneously accurate, and fast to obtain.

12.5 Experimental evaluation and discussion

12.5.1 Experimental results

We have tested our implementation of HMT-SLAM with two different datasets, both comprised of odometry readings and laser range scans taken in large planar scenarios containing several loops. The first dataset [Bla07a] was gathered by the thesis author at one of the campuses of the University of Málaga, and contains almost 5000 laser scans collected along a 2km path. The second dataset was recorded at the Edmonton Convention Centre (Canada) by Nicholas Roy, and is freely available online [How03].

For illustrating the experiments, the resulting HMT maps for both datasets are shown in Figure 12.8(a)–(b) as global maps (recall §12.3.3), taking the first area in each map as the reference. Videos describing these experiments can also be found online[4]. We overlap on the maps the most likely topological structure inferred by our approach to visualize the topological nodes and arcs. In contrast to global metric mapping with RBPF, our approach is able to maintain the uncertainty in all the relative poses between the areas, as can be seen with the (globally consistent) uncertainties represented in Figure 12.8(d). Recall that in RBPF-based global mapping without an adaptive number of samples, resampling steps eventually lead to a total loss of the represented uncertainty.

To compare the efficiency of other methods to our HMT-SLAM implementation, we have fed the same datasets into an efficient RBPF-based technique for (global) metric mapping [Gri07a]. The performances in computation time and memory requirements are summarized in Table 12.2. These values have been obtained for a 2.0GHz Pentium M (1Gb RAM) and for occupancy grid maps with a cell size of 12cm. It is noticeable

[4]Please, refer to the videos http://www.youtube.com/watch?v=i8lQdK6oRn0 and http://www.youtube.com/watch?v=PRHu5Py9GHo .

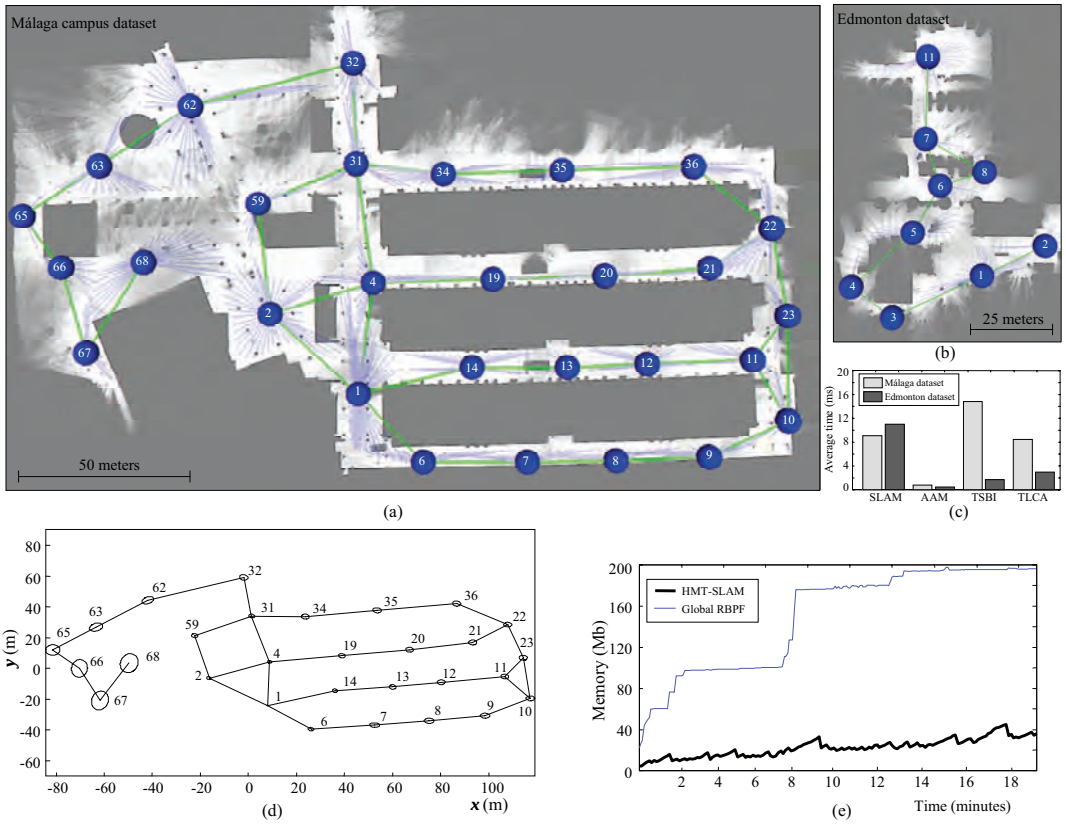


Figure 12.8: (a)–(b) The HMT maps generated from the Málaga and Edmonton datasets, respectively, shown as global maps relative to the first topological area (labeled as '1'). (c) The average time (per action-observation pair) taken by each of the processes in our HTM-SLAM implementation. (d) The globally consistent poses of each area for the Málaga map. The ellipses represent 95% confidence intervals magnified by factor 5 for clarity. (e) Comparison of the memory requirements over time between our approach and metric mapping with RBPF, both for the Málaga map.

Table 12.2: Comparison of Total Memory and Computation Time Requirements Between a Global Metric RBPF and HMT-SLAM

Method	**Málaga dataset**	**Edmonton dataset**
Global RBPF	197Mb, 103min	84Mb, 39min
HMT-SLAM	36Mb, 26min	28Mb, 8min

that HMT-SLAM outperforms global RBPF for both datasets. The improvement in the memory requirements follows from the fact that in our implementation of HMT-SLAM, each particle carries, apart from the topological path hypothesis, a metric map for the current area only, while the sub-maps of previous areas are kept in a compact form [Gri07b]. In the global RBPF, each particle carries a hypothesis of the whole metric map. Therefore, the storage efficiency of an HMT map in contrast to a global RBPF becomes more and more relevant for increasingly larger environments. This conclusion is supported by the evolution of the memory requirements over time for the Málaga dataset, plotted in Figure 12.8(e). Regarding the lower computation time of our approach, it is a direct consequence of the reduced number of particles (we use 15 of them). However, with only a few particles in our approach we can achieve a representation of uncertainty better than the one attainable by a global RBPF with a practical number of particles (e.g. around 100). This turns into more precise loop closures and a more reliable representation of uncertainty.

To get an approximate idea of the time consumed by each of the processes within our implementation (described in §12.4) we have measured their average time per action-observation pair in the datasets. As can be seen in Figure 12.8(c), the SLAM and AAM processes take roughly the same time in both datasets, while there are large differences for TSBI and TLCA. This reflects the fact that these latter two become more time consuming for a higher number of topological areas and potential loop closures, hence

they take much more time in the Málaga dataset than in the Edmonton dataset. Note that the RTL process has not been included in these experiments since they have been performed off-line. Nevertheless, it involves a negligible complexity in comparison to the rest.

It is interesting that for both datasets our method quickly converges to the correct topology after each loop closure. To visualize this, consider the first closure of a long loop in the Málaga dataset, where the robot leaves the area labeled as '14' and reenters the area '1' (refer to Figure 12.8(a)). Then, two hypotheses for the robot topological location are generated: a new, unexplored area ('15'), and an existing area that closes the loop ('1'). The intuitive idea of the expected behavior of our filter at this point is that the actual topological hypothesis will fit better to the robot observations, thus it will be assigned higher likelihood values within the particle filter that estimates the robot hybrid path s^t. Eventually, the correct hypothesis will become much more likely than the rest, which will be ultimately removed by resampling. This process is clearly observable in Figure 12.9, where we represent the effective sample size (ESS) [Liu96] of the filter and the marginal PMF of the topology over time. In these graphs we can see how before time t_1 there is a unique hypothesis for the topological position (area '14'), and next two possibilities are considered: an unexplored area '15' and a loop closure '1'. Since the loop closure hypothesis provides a much better explanation of the robot observations, its probability quickly increases, and after a resampling it becomes the unique hypothesis in the filter, definitively closing the loop.

12.5.2 Comparison to other large-scale approaches

It is also desirable to contrast how other hybrid methods perform in similar loop-closure situations. The Hierarchical SLAM approach reported in [Est05] computes the

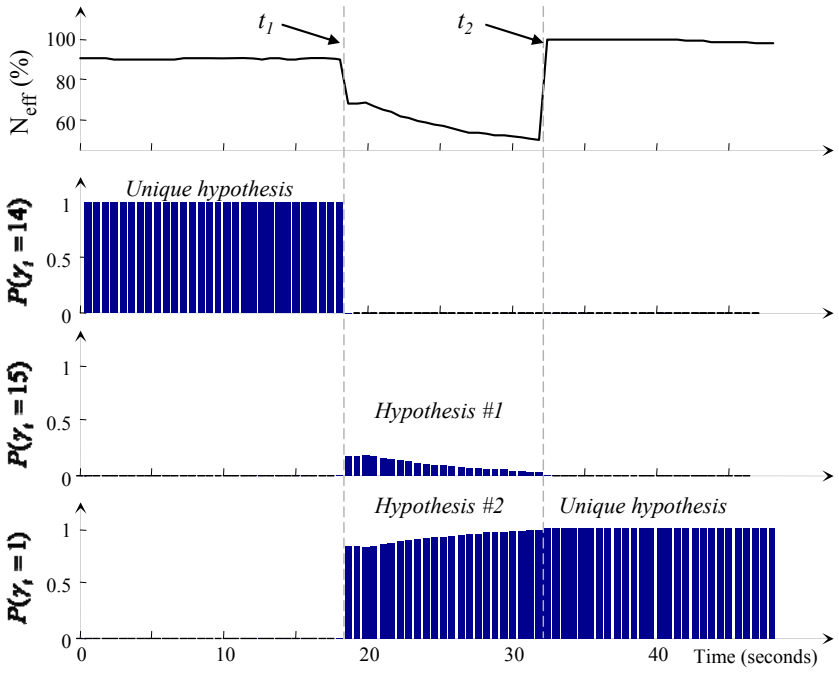


Figure 12.9: The evolution over time of some values in the mapping process while closing a loop between areas '14' and '1'. From the top to the bottom, the graphs represent the effective sample size (N_{eff}) of the particle filter, and the marginal PMF for the robot topological position evaluated for the areas '14', '15', and '1', respectively. Refer to the text for further discussion.

loop closure by least-square error minimization, thus it considers just one (metric) hypothesis for the closure. Although this may be enough in many situations, in highly repetitive scenarios it would be more advantageous to maintain multiple hypotheses until the closure becomes unambiguous. This is the case of the Atlas framework [Bos04], which considers the so-called *juvenile* hypotheses for closing loops. In that work, a juvenile hypothesis is promoted to *mature* when it performs much better than the rest, and until that point it is not allowed to modify its local map, that is, the treatment of hypotheses is purely heuristic. This heuristic approach is in contrast to our unified and mathematically grounded framework, where there are no distinctions between all the existing topological hypotheses, and all of them are allowed to modify their

corresponding local sub-maps (this, in turn, is required to perform sustainable accurate localization). Furthermore, since the Atlas framework is based on a hybrid robot pose (instead of hybrid robot *path*) it allows only one hypothesis for the robot metric pose within each sub-map, while under HMT-SLAM one can devise highly ambiguous environments where closing a long loop leads to a variety of hybrid path hypotheses, including the possibility of the robot being at the same area but with a multi-modal metric pose, and consequently with different local sub-maps.

12.5.3 Discussion

This chapter has presented a new approach for solving the problem of large-scale SLAM which consists of the unified estimation of a hybrid metrical and topological path of the robot throughout the environment. It has been demonstrated that this idea is supported by a probabilistic structure in the SLAM problem under plausible approximations. It has been addressed the probabilistic basis of a solution to HMT-SLAM in the form of sequential Bayesian filtering in the joint path-map space, which supports our approach as a promising step towards the natural integration of existing metric SLAM methods into high-level world representations, always within a Bayesian framework that manages spatial uncertainty more accurately and efficiently than previous metric and hybrid approaches. Additionally, an implementation of our ideas has been described in the form of a real-time/any-time system capable of providing an estimation of the robot pose and the map at each instant of time, giving more relevance to the computation of the metric robot pose, which may be required for navigation or manipulation purposes. Our work has been validated by experiments where relatively large HMT maps have been successfully built.

The introduction of HMT-SLAM gives rise to a number of interesting topics that

require future research, like the integration with other map types (i.e. landmarks), the simultaneous estimation of the HMT path of a team of robots, the consideration of alternative probabilistic estimation methods, or the use of appearance and high-level knowledge for localization and loop-closure. An especially interesting issue is that of solving the robot awakening problem, or global localization, within a large-scale and *partially known* environment (a problem that can be hardly dealt with existing metric or topological methods). This issue would be faced by any mobile robot operating in a realistic lifelong application. Under the perspective of our work, the problem is naturally cast as a special case of topological loop-closure. This means that, as a byproduct, HMT-SLAM will allow a robot to incorporate into an existing map new information gathered while performing global localization, and thus unifying SLAM and global localization. This latter point clearly deserves further research.

CHAPTER 13

CLUSTERING OBSERVATIONS INTO LOCAL MAPS

13.1 Introduction

As already mentioned in previous chapters, world models for SLAM can be classified broadly into metric ones, which use geometrical information [Gut99, Lu97a, Mon02a], and topological ones, which model the world as a graph whose nodes usually represent distinctive places [Bee05, Sog01]. As discussed in depth in Chapter 12, hybrid models that combine both types of information are a promising alternative when dealing with large and complex real robot environments. Typically, hybrid approaches attach local geometrical maps (suitable for metric robot localization) to the nodes of a graph-based world representation (suitable for task planning or reasoning) [Bos03, Est05, Thr96].

A crucial point then is to decide how to cluster the whole sequence of robot observations into local maps (sub-maps). From the different proposals reported in the literature, the following ones are of special significance (having been briefly discussed in Chapter 11): the Atlas framework [Bos03], where a new local map is started when localization performs poorly in the previous one; and, more recently, the hierarchical SLAM presented in [Est05], where sensed features are integrated into the current local map until a given number of them is reached. However, none of these provide a mathematically grounded solution for the problem or one that does not depend on strong human-provided heuristics that should be manually adjusted for each specific environment. Other interesting works pursue efficiency by exploiting the sparse nature of covariance matrices in the context of EKF-based SLAM [Pas02, Thr04, Wal07]. All these are based on the properties of covariance matrices within maps of *landmarks*, hence they are not applicable to other types of observations (e.g. raw laser range scans).

The approach discussed in this chapter consists of partitioning a graph-based representation of robot observations, usually called an *appearance-based representation* when using image sensors [Por06,Ryb03]. Here, the sequence of observations gathered by the robot (and the corresponding poses from which the robot takes each observation) are set as the nodes of an auxiliary weighted graph. By dividing this graph into disjoint clusters of highly connected nodes we can automatically determine a partition of the observed environment into "areas", which is required by hybrid approaches to mapping like HMT-SLAM. The semantics of these areas will be in general related neither to human concepts, such as rooms or a corridor, nor to operational needs. Rather, the distribution of the obtained sub-maps will be determined by the simultaneous visibility of landmarks from different robot poses: sensors with a wider field of view (FOV) or a larger sensing range will produce larger sub-maps since more overlap will be found

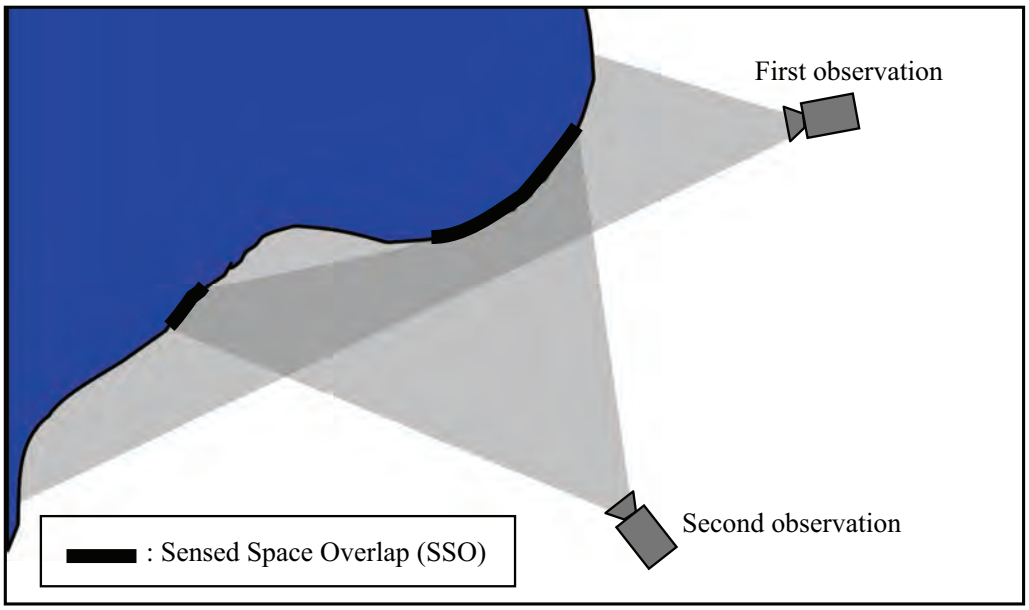


Figure 13.1: The sensed-space overlap (SSO) is a measurement of to what extent a pair of observations catch the same part of the environment.

between observations. This property that emerges naturally from the physical robot configuration is consistent with the ways of the biological world, where sensory capabilities definitively determine the spatial structure of world models.

An important contribution of the work presented in this chapter is the discussion of a new interpretation of the above process in probabilistic terms, hence providing a mathematical basis that justifies its usage in the context of HMT-SLAM: the resulting partitions will minimize a given measure of the relation between adjacent sub maps (as will be explained in §13.4) with the aim of obtaining sub-maps as closer to conditional independence as possible.

Given the mentioned auxiliary graph of robot observations, there are two critical issues regarding its partitioning: the computation of the arc weights, and the criterion for performing the partitioning itself. As introduced elsewhere [Bla06b], we propose

to set the weights according to the Sensed Space Overlap (SSO), a pairwise measurement between observations that reflects to what extent a pair of observations capture the same *entities* (points, landmarks, etc.) from the environment, as illustrated in Figure 13.1. Regarding the criterion for partitioning the graph, we follow previous works [Bla06b, Bla08d, Ziv05, Ziv06] that employ the minimum normalized-cut (min-Ncut), originally introduced in [Shi00]. The min-Ncut has the desirable property of generating *balanced* clusters of highly interconnected nodes, i.e. in our case clusters of observations that are very likely to represent the same part of the environment, under our definition for graph weights. Furthermore, it can be computed efficiently by means of an approximate solution based on spectral decomposition, which will be reviewed later on.

The remainder of this chapter is organized as follows. In §13.2 we review the spectral approach to graph partitioning for generic graphs. Next, we will derive expressions for computing the arc weights of the auxiliary graph for different kinds of sensors. In §13.4 we present the motivations and the formal support for partitioning a sequence of observations within a SLAM framework, and §13.5 explains how our method can be integrated into the HMT-SLAM framework. Finally, experimental results from real data validate the presented method.

13.2 Background on Spectral Graph Partitioning

In the following we review the definitions involved in the normalized cut, the basis for the *bisection* of a graph using the spectral approach, and how to extend it for generating any number of clusters.

13.2.1 Normalized Cut of a Graph

Let $G = \langle V, E, \Omega \rangle$ be an undirected, weighted graph, where V is the set of vertices or nodes and E is the set of weighted edges or arcs, using non-negative weight values; the symmetric $|V| \times |V|$ square matrix Ω is the weight matrix, where the element ω_{ij} is the weight of the arc between nodes i and j. According to the definition introduced by Shi and Malik in [Shi00], the *normalized cut* (Ncut) is a measure associated to the partitioning of V into *two* disjoint subsets A and $\bar{A}$, such as $A \cup \bar{A} = V$ and $A \cap \bar{A} = \varnothing$, defined as:

$$Ncut(A, \bar{A}) = \frac{cut(A, \bar{A})}{assoc(A, V)} + \frac{cut(A, \bar{A})}{assoc(\bar{A}, V)} \tag{13.2.1}$$

which in turn uses the standard definition of the *cut* between two disjoint sets of nodes A and $\bar{A}$:

$$cut(A, \bar{A}) \doteq \sum_{u \in A, v \in \bar{A}} \omega_{uv} \quad , A \cap \bar{A} = \varnothing \tag{13.2.2}$$

and of the *association* of two non-disjoint sets of nodes:

$$assoc(A, V) \doteq \sum_{u \in A, v \in V} \omega_{uv} \quad , A \subset V \tag{13.2.3}$$

The association of a given subgraph (A) with the whole graph (V) measures the intergroup "cohesion", that is, the connection "strength" between the nodes in A and those in the whole set V (including A again). Note that the definitions above fulfill:

$$assoc(A, V) = cut(A, \bar{A}) + assoc(A, A) \tag{13.2.4}$$

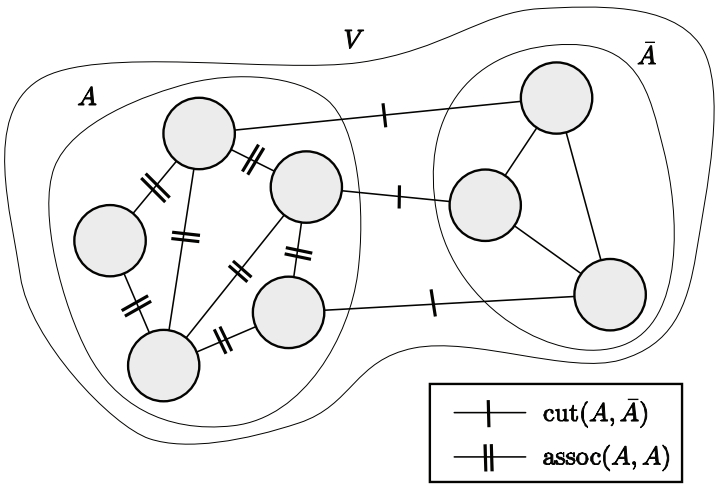


Figure 13.2: An example that illustrates the concepts of *cut* and *association* for a pair of sets of nodes. It can be seen how the cut involves only the arcs between two disjoint sets (A and $\bar{A}$ in this case), whereas the association takes into account all the arcs between the non-disjoint sets (A with itself in this example). Observe how the association of a set with the whole graph, i.e. $assoc(A, V)$, can be decomposed into the sum of its cut with the rest and the association with itself.

as Figure 13.2 illustrates with an example.

Partitioning graphs under the criterion of minimizing the *cut* value tends to generate groups of no practical utility for some applications, since they have the least connected nodes of the graph. It is of much more interest to get subgraphs with a balance between both, the intergroup and the intra-group cohesion, which is better achieved by minimizing the normalized cut (Ncut) defined in Eq. (13.2.1). Thus, the minimum normalized cut (min-Ncut) of a graph V is given by:

$$A = \arg\min_{A} Ncut(A, \bar{A}) \tag{13.2.5}$$

The range of possible values for the Ncut that result from this expression can be derived from Eq. (13.2.4). It can be deduced that, for the maximum value of the cut (which happens when the nodes in A are connected only to $\bar{A}$), the values of $assoc(A, A)$

and $assoc(\bar{A}, \bar{A})$ are zero, therefore:

$$assoc(\mathrm{A}, \mathrm{V})\big|_{\min} = assoc(\bar{\mathrm{A}}, \mathrm{V})\big|_{\min} = cut(\mathrm{A}, \bar{\mathrm{A}}) \tag{13.2.6}$$

Since the minimum value attainable from a cut is zero, corresponding to the case of no connections between A and $\bar{A}$, the minimum Ncut value is also zero. On the other hand, the maximum Ncut value is determined by the maximum values of each of the terms in the sum of Eq. (13.2.1). From Eq. (13.2.4) we see that the maximum value of each of these terms is:

$$max\left(\frac{cut(A,\bar{A})}{assoc(A,V)}\right) = max\left(\frac{cut(A,\bar{A})}{cut(A,\bar{A}) + assoc(A,A)}\right) = \frac{cut(A,\bar{A})}{cut(A,\bar{A})} = 1 \tag{13.2.7}$$

Thus, the Ncut provides a numerically well defined measure of the quality of a partition that falls within the range $[0, 2]$.

As discussed in the work by Shi and Malik [Shi00], finding the exact min-Ncut bisection is computationally intractable (an NP-complete problem), hence we follow their proposal for an approximate solution based on spectral decomposition, which leads to near-optimal cuts. Their method is summarized next for completeness, though it could be skipped by the reader since it is not necessary for getting into subsequent sections.

13.2.2 Spectral Bisection

Let x be the *bisection indicator vector* with dimension $N = |V|$, where each element x_i equals 1 or -1 depending of the node i falling into the group A or $\bar{A}$, respectively. Let d be the vector with the sum of the weights of the incident arcs for each node, that is

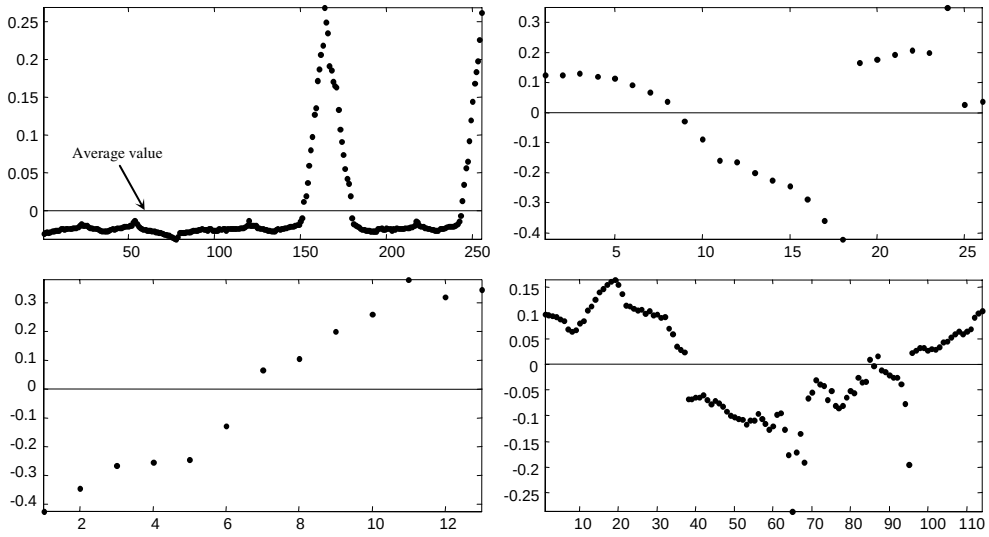


Figure 13.3: Four real examples of graph spectral bisections. The plots show the components of the eigenvectors (vertical axis) which are used to choose the bisection. The length of these vectors is given by the number of nodes in the graph (the horizontal axes represent node indices). The average of the eigenvectors (horizontal lines) is used as the bisection threshold.

$d_i = \sum_j \omega_{ij}$. We can build a diagonal matrix D with d as its diagonal. It can be shown that the min-Ncut problem can then be rewritten as:

$$\arg\min_x Ncut(x) = \arg\min_y \frac{y^t(D-\Omega)y}{y^tDy} \tag{13.2.8}$$

where $y = (1+x) - b(1-x)$, being 1 an $N \times 1$ vector of all ones, and

$$b = \frac{\sum_{x_i>0} d_i}{\sum_{x_i<0} d_i} \tag{13.2.9}$$

Ideally, the elements of vector $\mathbf{y}$ should take just two discrete values, since x_i can be either 1 or -1. However, if this condition is relaxed and $\mathbf{y}$ is allowed to be real-valued (this is the approximation of the approach), then Eq. 13.2.8 can be minimized by solving the generalized eigensystem:

$$(D - \Omega)y = \lambda Dy \tag{13.2.10}$$

where $L(V) = D - \Omega$ is a well-known term, namely the Laplacian matrix of the graph [Gol96]. The above equation can be rewritten as a standard eigensystem using $z = Dy$:

$$D^{-\frac{1}{2}}(D - \Omega)D^{-\frac{1}{2}}z = \lambda z \tag{13.2.11}$$

It can be shown that $z_0 = D^{\frac{1}{2}}1$, the eigenvector corresponding to the smallest eigenvalue in Eq. (13.2.11) ("the smallest eigenvector" from now on), is zero. Translating back this result to the original system in Eq. (13.2.10), we have that $y_0 = 1$ is the smallest eigenvector of Eq. (13.2.11). Since the fraction in Eq. (13.2.8) is a Rayleigh quotient [Moh91], and its eigenvectors are orthogonal [1], then both Eq. (13.2.10) and Eq. (13.2.11) are minimized for the next smallest eigenvector. Thus, we have arrived to the fact that solving the min-Ncut expressed in Eq. (13.2.8) can be approximated by finding the second smallest eigenvector y_1 of Eq. (13.2.10).

The only approximation assumed in the above derivation is that the components of the eigenvector y_1 will not take just two discrete values, but any real number. Obviously, this complicates the bisection criterion since we will need a threshold; still, in many situations there will be a clear distinction between the two clusters of nodes A and $\bar{A}$, as can be observed in the real examples shown in Figure 13.3. Three different criteria seem plausible for assigning each node a group:

1. Look at the sign of each component of the eigenvector,

[1] Since the Laplacian matrix $D - \Omega$ is positive semidefinite, $D^{-\frac{1}{2}}(D - \Omega)D^{-\frac{1}{2}}$ is symmetric positive semidefinite, thus its eigenvectors are orthogonal.

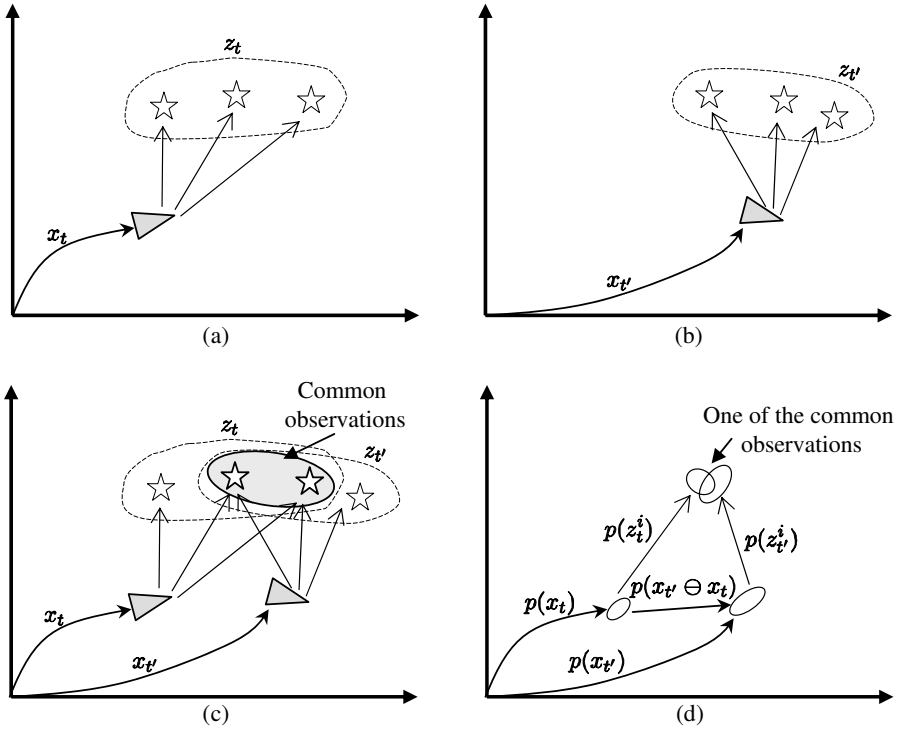


Figure 13.4: A graphical representation of the variables involved in computing the SSO for a pair of observations. (a)–(b) The robot pose and corresponding observations z_t and $z_{t'}$ at two time steps t and t'. (c) Some of the sensed elements (landmarks or points) may be common to both observations. (d) A representation of the probabilistic distributions taken into account when determining whether two elements from different observations correspond to the same map element or not. Refer to section 13.3.1 for further explanation.

2. take the mean value of the eigenvector as a threshold for the partition, and

3. sweep over the different threshold values looking for the minimum Ncut.

The second one is the criterion we have used in our implementation due to its compromise between efficiency and good results.

13.2.3 Partitioning graphs into k groups

The method presented above provides a solution to the graph bisection problem. However, this method must be generalized in our case for dividing a graph into a variable

number of subgraphs. An easy and effective way of achieving that is to recursively apply bisection to any of the generated subgraphs as long as two clearly differentiated groups can be obtained.

The resulting min-Ncut value for a given bisection is a well-grounded measure of the goodness of the cut, with values in the range $[0, 2]$ that measures the inter-group cohesion of the resulting subgraphs, inversely scaled by the intra-group cohesion. Values close to zero indicate almost no connection between the groups (a good partition), while values near 2 indicate that the groups are more strongly connected to each other than with themselves (the partition should not be done). Therefore, an intermediate value must be established as a threshold $0 < \tau < 2$ to decide whether the bisection should be accepted or not. This value is typically chosen heuristically [Shi00, Vek00], and in practice a constant value in the range $[0.2, 1]$ will give good results. The complete procedure for k-ways partitioning is summarized in Algorithm 3.

Algorithm 3 recursive_partition $G \rightarrow \{P\}$

1: $\{A, B\} \leftarrow spectral\_bisection(G)$
2: $N_{cut} \leftarrow compute\_ncut(G, \{A, B\})$ *// N-cut of the candidate bisection*
3: **if** $N_{cut} < \tau$ **then**
4: *// Accept the cut: do recursive call*
5: $P \leftarrow \{spectral\_bisection(A), spectral\_bisection(B)\}$
6: **else**
7: *// Do not accept the cut: G cannot be divided anymore*
8: $P \leftarrow G$
9: **end if**

13.3 The overlap-measuring function SSO

In this section we address the problem of assigning weights to our graph of observations. To this purpose, we introduce the Sensed Space Overlap (SSO), a pairwise similarity measure for observations that reflects how much space in common a pair of observations sense.

Let the map variable m be comprised of the set of variables that correspond to each individual map element, such that $m = \{m^1, ..., m^i\}$. Similarly, the observation z_t taken at a certain time step t will be considered as composed of the observations of individual map elements, that is, $z_t = \{z_t^1, ..., z_t^k\}$. The nature of these "map elements" is not relevant for the generic definition of the SSO[2]. Furthermore, we will denote the set of map elements sensed in a given observation z_t as $M(z_t)$. Put mathematically:

$$M(z_t) = \left\{m^i : z_t^k \text{ observes } m^i, \forall\ k\right\} \tag{13.3.1}$$

At this point, we can define the SSO function in general for any pair of observations z_a and z_b as the ratio of commonly observed map elements relative to the overall number of observed elements. Using the notation defined above, the generic expression for the SSO is:

$$SSO(z_a, z_b) \doteq \frac{|M(z_a) \cap M(z_b)|}{|M(z_a) \cup M(z_b)|} \tag{13.3.2}$$

where $|\cdot|$ stands for the cardinal (size) of a set. The SSO can be particularized for each kind of sensor. Next we derive expressions for two kinds of sensory data: landmarks and range scans. A similar definition that could be employed for monocular images has been proposed in [Ziv05].

[2]They would be cells for an occupancy grid map, individual features in a landmark-based map, etc.

13.3.1 SSO for landmark observations

We are interested in computing the similarity $SSO(z_t, z_{t'})$ between the observations z_t and $z_{t'}$ taken at time steps t and t' from the robot poses x_t and $x_{t'}$, respectively. In the case of landmarks, each observation z comprises a set of individual features $\{z^i : i = 1..k\}$, one for each sensed map element. An individual feature z^i represents the observed spatial position of the i'th landmark relative to the robot pose x at the corresponding time step. All the involved variables are represented graphically in Figure 13.4(a)–(b) for clarity. The only problematic step in computing the SSO here is determining the number of matches from common elements between the pair of observations z_t and $z_{t'}$, as those shown in the example in Figure 13.4(c). Concretely, to decide if a pair of individual features z_t^i and $z_{t'}^j$ correspond to the same map element, we have to check whether the following equality holds:

$$x_t \oplus z_t^i = x_{t'} \oplus z_{t'}^j \tag{13.3.3}$$

being $\oplus$ the pose composition operator [Smi88].

Within a probabilistic SLAM framework there is uncertainty in measures and poses, thus we do not know the exact value of the variables, but only their associated probability distributions. Therefore, we can only decide for correspondences within a given certainty bound. This can be visualized with the uncertainty ellipses shown in Figure 13.4(d), where it can be seen how the position of the same landmark is represented by different (although overlapping) Gaussians for each of the observations.

If we rewrite Eq. (13.3.3) as [3],

[3] Using the matrix form of Eq. (13.3.3) in homogeneous coordinates, $\mathbf{X}_t \mathbf{z}_t^i = \mathbf{X}_{t'} \mathbf{z}_{t'}^j$, we obtain $\mathbf{X}_t^{-1} \mathbf{X}_{t'} \mathbf{z}_{t'}^j - \mathbf{z}_t^i = 0$, which is stated in Eq. (13.3.4) using pose composition operators.

$$\epsilon = (x_{t'} \ominus x_t) \oplus z_{t'}^j - z_t^i = 0 \tag{13.3.4}$$

with $\ominus$ being the inverse pose composition operator (see appendix A), we can then state the probability *density* of actually having a correspondence through $p(\epsilon = 0)$. If we assume that the probability distributions of both the robot pose and the landmarks can be appropriately approximated by Gaussians (which is acceptable in an HMT-SLAM framework since coordinates into local maps are always relative to a near reference [Bla08d, Bos03, Tar02]), then $p(\epsilon)$ can also be approximated by a Gaussian by first-order uncertainty propagation (we omit the straightforward calculations). As a result, $p(\epsilon)$ would be centered near the origin if there is a correspondence, or far away otherwise. A robust criterion for deciding the correspondence is therefore to compute the Mahalanobis distance from the origin to the mean of the Gaussian, and to accept the correspondence if the distance is below, for example, a value of 3 (which represents a 99.7% confidence interval) – a procedure that is, in fact, a chi-square test (see §2.3.3).

Notice that we are assuming that all the landmarks are indistinguishable and the matching must be determined solely from spatial information, but there are some situations where landmarks have some sort of descriptor, which can then be integrated into the calculation of the Mahalanobis distance. In concrete, for the experiments shown at the end of this work, we have used SIFT [Low99] visual descriptors in addition to the spatial distance, as described in more detail in [Mor07].

13.3.2 SSO for range scans observations

In principle, we could apply the same process as in section 13.3.1 to compute the SSO of a pair of observations comprising raw range scans. However, there are subtle differences in the nature of the sensory data which make desirable the introduction of a

slight modification: due to the discrete set of scanning directions it is very unlikely that exactly the same *point* (not landmark) is sensed while scanning the environment from different poses. We present a solution for accounting for this fact when considering uncertainties. That is the only difference with the process described above for landmarks.

If we denote as $\mathbf{C_t^i}$ the covariance of the 2-d point $x_t \oplus z_t^i$ (please, refer to Figure 13.4(d)), we can model the uncertainty due to the discrete sampling of the environment by summing an additional term σ_f to the diagonal of the covariance:

$$\mathbf{C_t^i} = \mathbf{J} \begin{pmatrix} \mathbf{\Sigma_p} & 0 \\ 0 & \mathbf{\Sigma_s} \end{pmatrix} \mathbf{J}^t + \begin{pmatrix} \sigma_f^2 & 0 \\ 0 & \sigma_f^2 \end{pmatrix} \tag{13.3.5}$$

where Σ_p and Σ_s are the covariances of the robot pose and sensor measurement, and $\mathbf{J}$ is the 2×5 Jacobian matrix of the pose composition operator. Since the spatial uncertainty due to the discrete sampling of surfaces is proportional to the sensed range (r) at each scanning direction, the value σ_f can be set to $r\beta$, being β a constant of the order of the discrete angular steps between the scan ranges.

13.3.3 An Example

It is illustrative at this point to consider an example to show how our overall method works for partitioning the graph of observations shown in Figure 13.5 (simulated raw range scans in this case). Here, the graph is firstly divided into the groups $\{G1\}$ and $\{G2, G3\}$ in the first execution of the partitioning algorithm. Going on recursively, the latter group is partitioned again due to its low Ncut value. The so resulting groups $\{G2\}$ and $\{G3\}$ are no longer bisected since the corresponding minimum Ncut values are above the threshold (set to 1 in this example), i.e. it is better not to separate the observations between each group. Notice that the final observation groups do

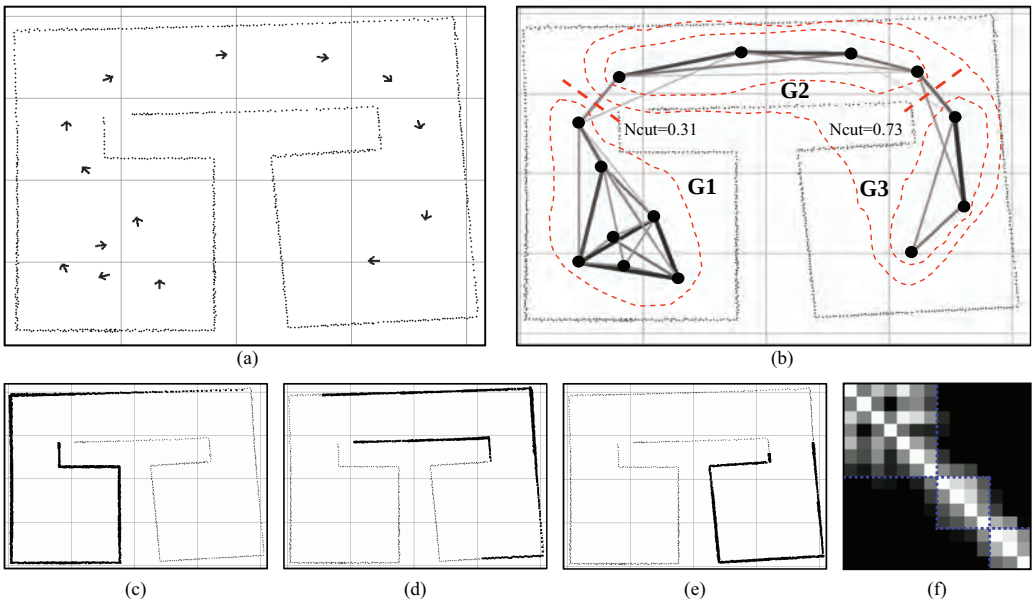


Figure 13.5: An illustrative example of the graph partitioning method applied to a 2-d laser map. (a) The global map obtained from 14 observations – arrows indicate the poses where observations were taken from. Notice that the map presents some orientation errors. (b) The auxiliary graph of observations. Each node contains the sensed space data (scan), and an estimate for its pose. The darker the arc, the higher the SSO between the observations. The observation graph is recursively partitioned into three groups: firstly, it is divided into two groups $\{G1\}$ and $\{G2, G3\}$, then, the latter is partitioned again because it has a minimum Ncut below the threshold. The local maps obtained from these groups are shown in (d), (c) and (e), respectively. In (f), the weight matrix of the associated graph in (b) is shown as an image with dotted squares for the three partitions.

roughly correspond to each of the natural "rooms" that can be observed in the figure, although, as commented in the introduction, our partitioning method does not aim for "human-like" semantics, but for a "subjective" perception of the world by the robot.

13.4 Theoretical support for HMT-SLAM

In this section we derive a justification for the usage of the SSO as a metric within a min-Ncut partitioning in the context of the hybrid SLAM framework presented in Chapter 12.

Following the standard notation in the SLAM literature, the ultimate goal of any probabilistic localization and mapping method is to compute the joint posterior density of both the robot path $x_{1:t}$ and the map m given the sequence of observations $z_{1:t}$ up to time step t:

$$p(x_{1:t}, m|z_{1:t}) \tag{13.4.1}$$

where the robot actions have been dropped for clarity. Stated as a sequential Bayesian estimation problem, the statistical structure of the variables is the one illustrated in Figure 13.6(a) as a dynamic Bayesian network. A critical issue in this model is that any observation z_t obtained by the robot depends on the *whole* map, represented by m. Although this condition is rigorously true, in practice the observations capture only a limited part of the map. Addressing the SLAM problem through our HMT-SLAM approach is indeed based on this observation.

In HMT-SLAM, the map m is divided into a set of metric sub-maps which can be estimated from (ideally) conditional independent sequences of observations. Here we discuss why the min-Ncut using the SSO for arc weights is a good choice for generating

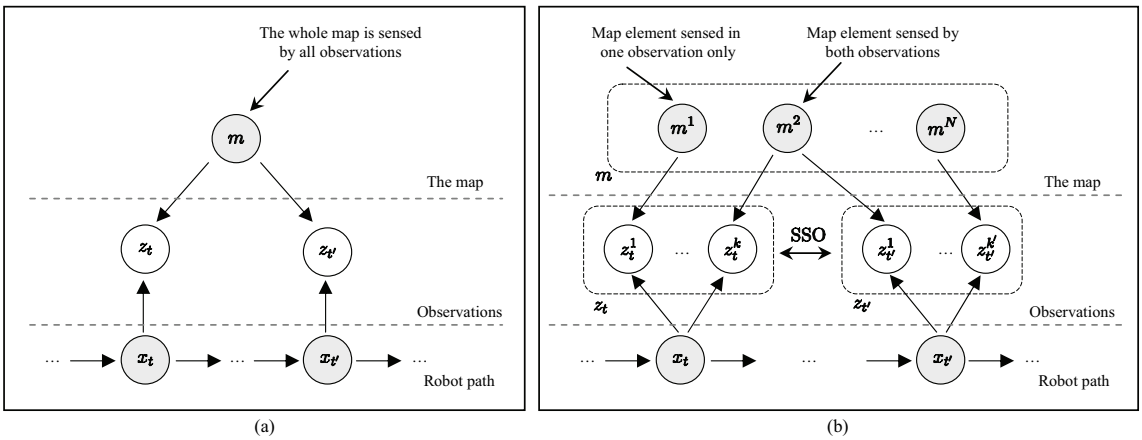


Figure 13.6: (a) The common structure of the SLAM problem as a dynamic Bayesian network. (b) The same model if we split up the map and the observations into its separate variables for the individual map elements. Under this perspective, the SSO can be defined as the ratio of common map elements sensed by a pair of observations. As discussed in the text, this quantity is related to the "degree of independence" between the involved variables.

these sub-maps. The following reasonings are also applicable to other hybrid (or "hierarchical") approaches to SLAM [Bos03, Est05], since the approximations introduced by any approach that divides the map into sub-maps are more negligible as the observations between the different clusters become closer to conditional independence (given the robot path).

As illustrated in Figure 13.6(b), the SSO captures the fact that there may be some map elements m^i sensed by just one of the two observations, and other map elements sensed by both of them. It is straightforward to verify from Eq. (13.3.2) that the SSO function gives values in the range $[0, 1]$, suitable to be used as weights in the arcs of the auxiliary observation graph: observations sensing exactly the same map elements are assigned a value of 1 and observations coming from totally different parts of the environment have a SSO value of 0.

Consider now the particular case of a null *cut* value between two clusters of observations A and $\bar{A}$ using the SSO as the arc weights, which in turn implies a null Ncut

value (see Eq. (13.2.1)), that is:

$$cut(A, \bar{A}) = \sum_{a \in A, b \in \bar{A}} SSO(z_a, z_b) = 0 \tag{13.4.2}$$

Since the SSO is non-negative, it follows that:

$$SSO(z_a, z_b) = 0 \quad \forall (a, b) \in A \times \bar{A} \tag{13.4.3}$$

which, given Eq. (13.3.2), implies that:

$$M(z_a) \cap M(z_b) = \varnothing \tag{13.4.4}$$

This relation can also be proved in the opposite direction: two sets of observations that do not contain any common map element have a null cut (and Ncut) value. In other words, using the min-cut or the min-Ncut criteria for partitioning is *equivalent* to finding sets of observations with the least map elements in common. Our choice for the min-Ncut (rather than the min-cut) criterion is due to its desirable property of producing more balanced groups (refer to [Shi00]), as discussed above.

To sum up, for the case of a null Ncut value we can rigorously factor the SLAM problem into statistically-independent estimations. In practice, it is virtually impossible to obtain such strict independence between difference areas of a large map. In the scope of HMT-SLAM, we propose to settle a threshold value τ for the maximum admissible Ncut value in order to partitioning the map into suitable areas, whose value coincides with that employed in the recursive algorithm in Algorithm 3. Note that this threshold value does not depend on the kind of sensors employed since Ncut values will represent relative SSO values.

13.5 Sequential clustering within HMT-SLAM

In the previous sections we have stated the problem of partitioning a sequence of observations as an off-line process, assuming a static and complete sequence of observations and associated robot poses. In the following we describe the issues raised when our method is applied to online SLAM.

Firstly, we must remark that there may exist several ways of integrating the partitioning technique into our HMT-SLAM framework. In concrete, we propose here to take the sequence of the last robot observations and partition it to check whether the robot has entered into a new area or not at each time step. More generally, the recursive partitioning algorithm would reveal the different topological areas in which the robot observations nearby its current position can be grouped into.

Within this context, a new node containing the last observation and the current probabilistic estimate of the robot pose is attached to the auxiliary observation graph for each time step of the SLAM algorithm. The only difference to an off-line (batch) version of our method lies in the computation of the weight matrix Ω, which is to be built sequentially as new nodes are added to the graph. Let Ω_t denote the weight matrix for the sequence of observations gathered up to time step t. For each new observation z_t this matrix can be updated just by expanding the previous one (Ω_{t-1}) with a new row and a new column:

$$\Omega_t = \left(\begin{array}{c|c} \Omega_{t-1} & \omega_{1:t-1,t} \\ \hline \omega_{1:t-1,t}^T & 0 \end{array}\right) \tag{13.5.1}$$

Since the weight of the reflexive arc that connects each node with itself is not employed in the calculations of the min-Ncut, the diagonal elements in the Ω matrix can be

set arbitrarily to zero. Each update of the weight matrix involves the evaluation of the SSO for $t-1$ pairs of observations, corresponding to the column $\omega_{1:t-1,t}$ in Eq. (13.5.1). Thus, updating the Ω matrix at each time step has a computational complexity that increases linearly with t. After updating the matrix, a bisection eigenvector must be computed (recall section 13.2.2), which can also be achieved in $O(t)$ by applying the Lanczos algorithm [Gol96]. To sum up, the overall complexity remains linear with the number of previous observations. This growth in complexity over time is not a problem as long as eventually the robot moves to a different area and a new matrix is created. Thus, in practice there is an upper bound to the size of this matrix, although it will depend on the specific structure of the environment and on the robot path.

13.6 Experimental evaluation

We firstly provide some statistical results aimed to compare our method to other previous proposals, and next we show some typical partitions obtained for an indoor scenario for several combinations of sensors[4].

13.6.1 Statistical experiments

As discussed in section 13.4, separating the map into clusters will incur in approximation errors for most practical situations. In order to compare the loss caused by our method to other alternatives, we will define a measure of how much information is lost by performing any arbitrary partitioning of a map. The context for this comparison is EKF-based SLAM [Dis01] rather than the RBPF-based approach employed in the next section. The reason for using an EKF with a map of landmarks here is that the cross

[4]The datasets and C++ source code for these experiments are available at http://www.mrpt.org/papers/.

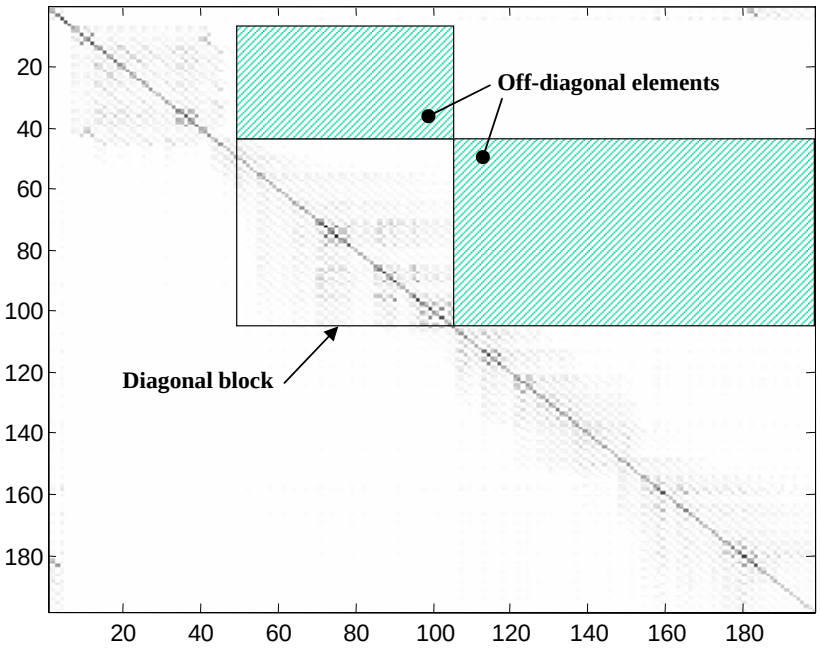


Figure 13.7: The information matrix of an EKF solution to SLAM represents the shared information between landmarks in the map. Partitioning a map implies assuming conditional independence between different sub-maps: the matrix elements out of the diagonal block matrices are forced to zero. We employ this matrix to measure the information loss due to different partitioning methods.

covariances between map elements are explicitly kept in the filter covariance matrix, whereas in grid-mapping with a RBPF they are not available. The covariance matrix is important for our purposes since its inverse, the *information matrix*, represents the amount of shared information between the different landmarks in the map. Under a hybrid approach to SLAM, where the aim is to partition the map into statistically independent clusters, the information is maintained in block diagonal sub-matrices only, while the off-diagonal elements are discarded (i.e. assumed to be zero); this is illustrated with an example matrix in Figure 13.7. Actually, approaches to SLAM based on a Sparse Extended Information Filter (SEIF) explicitly set to zero some entries of the matrix (which were not zero previously), therefore assuming a certain loss of information [Thr04, Wal07]. We must remark that the method presented here does not rely

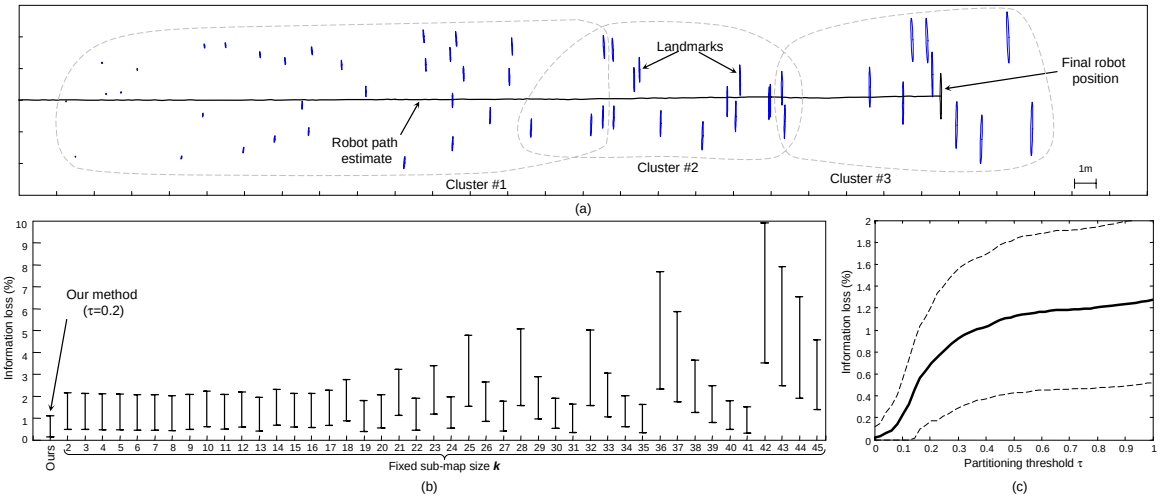


Figure 13.8: Results for the statistical analyses of the information loss due to map clustering. (a) The final state of an EKF after one of the simulations. Observations are grouped (in grey) according to the three clusters produced by our method in this particular run. Recall that our method clusters *observations*, not *landmarks*, hence that some landmarks belong to more than one group. (b) The average and 68% confidence interval for the information loss e, calculated for both our method (the left-hand value) and for fixed size sub-maps comprised of k observations each. (c) The information loss and the 68% confidence interval for different thresholds τ in our method.

on the information matrix as those based on a SEIF, hence its applicability to other kinds of mapping frameworks such as grid mapping with RBPFs.

We propose to measure the effects of forcing conditional independence (i.e. partitioning the map) through the information loss ratio e, defined as:

$$e = \frac{\sum_{(i,j):C(i)\cap C(j)=\varnothing} |H(i,j)|}{\sum_{a,b} |H(a,b)|} \tag{13.6.1}$$

Here $H(i,j)$ stands for the information matrix entries for landmarks i and j, and $C(i)$ represents the set of clusters which the landmark i belongs to. Put in words, the loss ratio e is proportional to the sum of all the information matrix elements out of the block diagonal matrices for each cluster in the partition (please refer to Figure 13.7).

We have carried out simulations by running 150 times a simple EKF [Bla08b] for

a planar path consisting of a straight trajectory of 50 meters through an environment with 60 point features uniformly distributed, placed at random positions for each run. The final state of the filter after one of the runs is shown in Figure 13.8(a), including the three clusters in which our method divides the observations in this case. Loops in the robot path have been intentionally avoided to keep the experiment simple and to obtain generic results independently of the implementation framework.

The first result from the simulations is the comparison of the average information loss (e) for the different methods for partitioning the sequence of observations, summarized in Figure 13.8(b). The left-hand value corresponds to our proposal, in this case using a fixed value of 0.2 for the recursive bisection threshold τ. It can be seen that the largest part of the confidence interval for e is below 1%, which is in contrast to the other alternative method, which consists of starting a new map after a fixed number k of observations, with k varying from 2 to 45. Although the loss of information is not drastic (less than 3% for most values of k), it is clear that our method not only implies a more reduced approximation error, but it is also more predictable as revealed by the lower variance of e. We must note that starting a new sub-map every fixed number of features was proposed in [Est05], but other heuristics, such as starting a new sub-map when localization performs poorly [Bos03], can be also ultimately expected to start new maps at a regular rate if landmarks are distributed uniformly through the environment.

A second statistical result is the characterization of our approach with respect to its parameter – the threshold τ. As expected, lower values of τ lead to lower approximation errors e, as represented in Figure 13.8(c). Though this may seems to suggest to always use a threshold close to zero, in practice a compromise should be found between admissible errors and the size of the sub-maps, which grows as τ

decreases. It is noteworthy that the expected error for $\tau = 0$ should be zero since submaps will be created only when no observation senses two landmarks from different clusters. Instead, our simulations give an average information loss of $e = 0.022\%$ (negligible, but not null). The reason of this result is that we integrate odometry in the simulations, which is known to create correlations between landmarks even when they have been not observed simultaneously (refer to [Wal07] for a enlightening discussion on this topic).

13.6.2 Partitioning a real indoor map with several sensors

To demonstrate the partitioning obtained by our method in a real setting we have applied it to a sequence of observations gathered by one of our mobile robots, which is equipped with a front SICK laser scanner, a rear HOKUYO laser scanner, and a stereo camera pointing forward. SIFT image features are extracted from the stereo images at each time step to generate the set of 3D features (more details can be found in [Mor07]) that we will consider landmark-like observations. To reveal the differences in the obtained partitions for each kind of sensor and their possible combinations, we have performed the partitioning four times: using the front laser scanner only, both laser scanners (providing a field of view of almost 360° – except for small lateral dead angles), visual landmarks only, and the three sensors simultaneously. When several sensors are combined into the same observation simultaneously, the corresponding SSO is computed by averaging the individual SSO functions, as can be easily derived from the definition in Eq. (13.3.2). The effects are discussed below.

The results are summarized in Figure 13.9, where each row represents one set of sensors, and the left column shows the resulting groups of observations on an occupancy grid map of the environment (visual landmarks are not shown for clarity). The

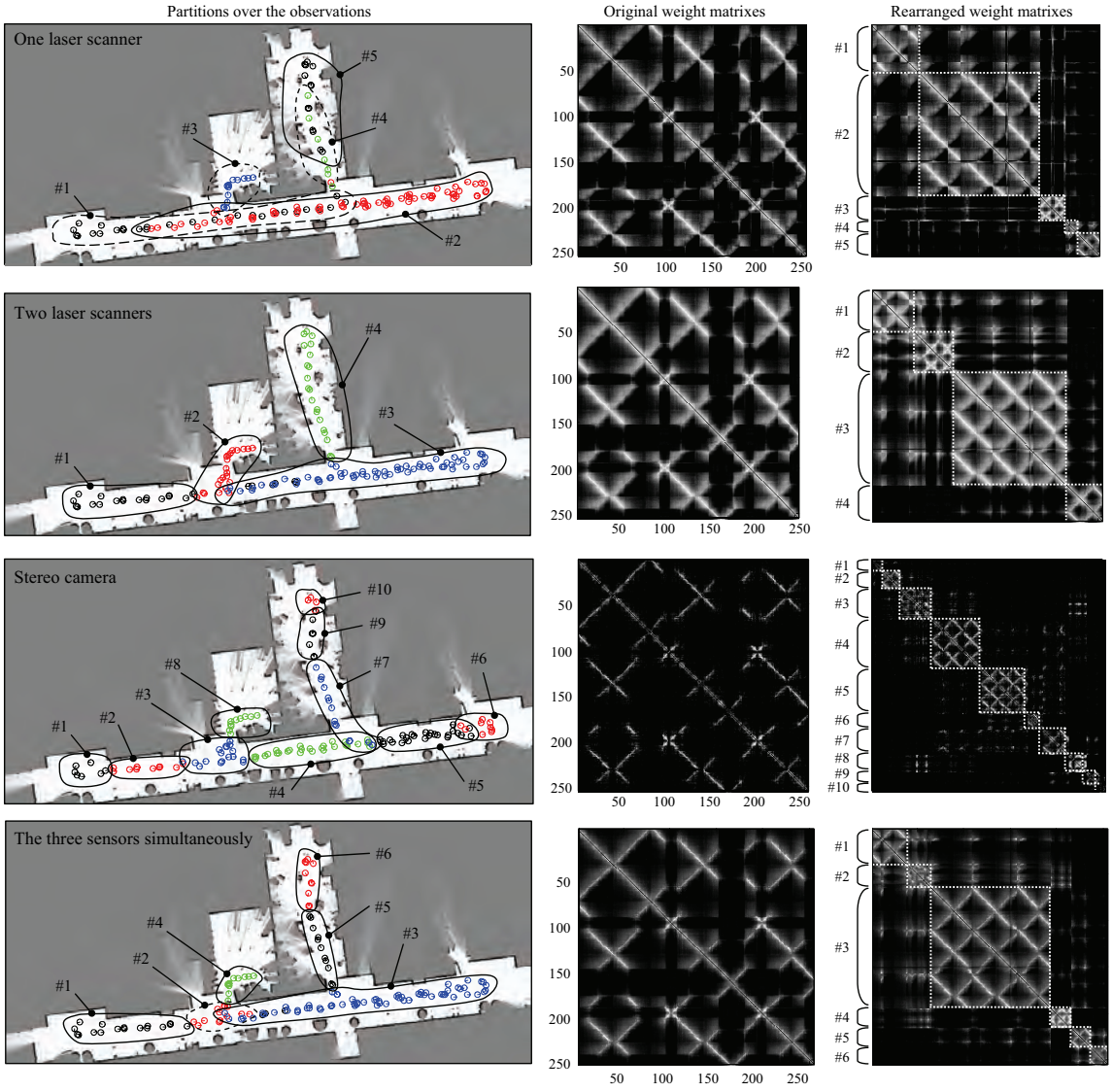


Figure 13.9: Results from partitioning a map gathered by a real mobile robot. The left column shows the resulting clusters of observations obtained by the proposed method, while the middle and right columns contain the weight matrices Ω before and after rearranging the elements according to the partitioning, respectively. It can be seen how after the rearrangement most of the high values in Ω are within the diagonal blocks. The rows of graphs present the results for a different set of sensors: one 180° laser scanner, two laser scanners (bottom row), a stereo camera, and the three sensors simultaneously.

middle column shows the final SSO (arc weight) matrix for the time-ordered sequence of observations through the environment. Since the robot revisits the same places several times, it is remarkable that many off-diagonal elements contain high SSO values, i.e. they correspond to close areas. In the right column in the figure we can observe the rearranged matrices, built up by making the observations within the same cluster to have consecutive indexes: after clustering, the output matrix should be block diagonal ideally. In the figure we can see how the rearranged matrices for the experiments are clearly not block diagonal, but most of the non-zero elements are approximately contained within the block diagonal matrices. The elements outside these diagonal blocks are the information that would be lost in the hybrid SLAM approach.

It is interesting to note the differences in the partitions obtained from the different sensors. Firstly, for the case of just one 180° FOV laser scanner (the top row in the figure) there exists overlap between different detected areas. For example, the groups #1 and #2, or the groups #4 and #5. However, actually each overlapping group contains observations with opposite robot headings, that is, since the FOV is 180° there is almost no overlap between the observations taken by the robot going in one direction and in the opposite. This does not occur in the second case, when two laser scanners are considered simultaneously covering almost 360° around the robot. For these sensors, we obtain the clustering closest to the "human" concept of rooms, with almost no overlap between the resulting areas. This is in contrast to the results from a stereo camera (third row in Figure 13.9), with a narrow FOV of roughly 65°. We preprocess the images from the camera to obtain a set of 3D landmarks, which we consider as the camera observations themselves (please refer to our previous work [Mor07] for a description of the process). More different areas are detected in this case (10 areas), whereas they were just 4 for the case of two laser scanners. The reason is that the narrow FOV leads to many groups of a few observations each, specially if the robot

rotates.

In the case of using all the three sensors (two laser scanners and the stereo camera) we obtain the clusters shown in the bottom row of the figure. Here the weight matrix Ω is the average of those from the individual sensors (notice how this matrix, in the central column of the figure, is a mixture from the matrices at the second and the third row). We obtain 6 areas, which is an intermediary value between that obtained from the two laser scanners and the camera independently. Therefore, mixing sensors with largely different FOVs (360° vs. 65°) could be seen as having one single sensor with an intermediate FOV, a natural consequence of computing SSO by averaging over the different sensors.

Finally, we can observe in the rearranged weight matrices (at the right in Figure 13.9) how most of the SSO high values are within the diagonal blocks corresponding to the clusters computed by our method. However, a few values are left out of these blocks. Within a hybrid SLAM method this would mean that the information out of the block-diagonal approximation of this matrix would be lost.

13.6.3 Discussion

Spectral techniques have been employed for efficient graph partitioning in the generation of sub-maps for hybrid SLAM frameworks. Through the introduction of the SSO function, we have provided a general formulation for partitioning sequences of observations from different sensory data, illustrated with both range scans and landmarks. An important contribution of this work is the discussion of a probabilistically grounded interpretation of the usage of min-Ncut and the SSO function to probabilistic hybrid SLAM. We have provided a statistical analysis of our method compared to other alternatives, as well as analyzed how the use of sensors with different FOVs affects the

resulting clusterings obtained from a real data set. It has been thus verified how our approach produces robot "subjective" local maps.

CHAPTER 14

GRID MAP MATCHING FOR TOPOLOGICAL LOOP CLOSURE

14.1 Introduction

As discussed in Chapter 12, an important operation within HMT-SLAM is detecting whether two local maps correspond to the same physical place and, in that case, to compute the relative transformation between those maps, that is, detecting *topological loop closures*. Solving loop closures in a hierarchical framework, which is the purpose of the method presented in this chapter, implies coping with a number of hurdles such as noise in the robotic sensor, ambiguity (different parts of the environment can be indistinguishable) and dynamic scenarios (the map of an area may change over time).

A method for matching grid maps is presented here as part of a more complex

method for the case of "hybrid" [1] representations of local maps where both occupancy grids and point maps are maintained. As described in [Nie04], this approach has a number of advantages since these kinds of maps complement each other and their maintenance only requires updating both maps simultaneously with the same sensory data. Nevertheless, it must be remarked that the presented method could be applied to grid maps only.

Correspondingly to this hybrid representation that uses both point and grid maps, the overall approach for aligning a pair of local metric maps consists of two differentiated steps: (i) the grid maps are firstly matched without any a priori information, then (ii) the point maps help to refine the matching. Our discussion will preeminently focus on the first step, the *grid-to-grid matching*, since the registration of point maps is a well understood topic with efficient solutions such as ICP [Bes92] or 3D-NDT [**?**]. Furthermore, this second step only has to refine an estimation already close to the real solution, while the grid-to-grid matching has no such advantage and thus poses a far more challenging problem.

In this chapter we propose to estimate the transformation between a pair of grid maps by registering the corresponding *map images*, the grayscale images resulting from interpreting grid cells as pixels and occupancy probabilities as gray levels. Since in robotic applications we can select the grid cell size, we focus on matching maps with identical cell sizes only. In this case, a pair of maps can be only related by a 3-d transformation, or *pose*, fully determined by a 2-d translation plus a rotation, disregarding scale changes.

In general, image registration techniques can be classified into those based on intensity and those based on the extraction of interest points – refer to [Zit03] for an

[1] We use here the original term "hybrid" as introduced in [Nie04], but this is not related at all to the term "hybrid" used to designate our hierarchical approach to large-scale slam (HMT-SLAM).

extensive review. Although the former approach has been already applied to grid map matching [Gut99], there is no previous work on grid-matching based on feature extraction, which is known to be more efficient computationally and therefore more appropriate for being integrated into real-time mapping frameworks.

Our approach represents an important contribution due to its robustness for finding the transformation between map images in the form of a Sum of Gaussians (SOG). This probabilistic representation allows coping with multiple hypotheses and therefore to consistently integrate the method into robotic mapping frameworks, most of them based on probabilistic Bayesian inference [Thr05]. This probabilistic approach is in contrast to previous works on robust image registration based on vote-counting in the space of transformation parameters [She99]. Within mobile robotics, Duckett and Nehmzow [Duc01] reported a method very similar to ours, which also obtains a SOG for potential matches between grid maps. However, their work assumes an accurate knowledge of the absolute orientation of the robot (i.e. it should be equipped with a compass), hence our proposal has a broader applicability to practical situations.

The work presented in this chapter is also related to research in multi-robot mapping, since the map merge problem there can be seen as a special instance of the detection of loop closures in single robot mapping. In that field, a method with a similar purpose to ours has been reported in [Bir06], but it does not consider the possibility of multiple hypotheses in the map merging, and a rough comparison of typical execution times has revealed that our method is about 100 times faster.

In the next section it is provided an overview of the method. A benchmark of feature point detectors and descriptors is provided in §§14.3–14.5. The robust matching method, discussed in §14.6, requires a Gaussian model for the optimal rigid transformation for subsets of correspondences, which is derived in §14.7. Finally, experiments are presented with maps from four publicly available datasets.

14.2 Overview of the method

The overall method is summarized in Figure 14.1. Firstly, map images are preprocessed to soften out the irregularities commonly found in grid maps, which can be seen as high-frequency noise. Interest points (*features*) are then detected in the filtered images and descriptors computed to model their surroundings. Obviously, the choice of a particular interest point detector and descriptor will determine the performance of our whole method. After comprehensive experiments, which are reported in §§14.3–14.5.2, it has been determined that either the Harris [Har88] or the Kanade-Lucas-Tomasi [Luc81,Shi94] detectors, in combination with a descriptor consisting of a circular patch centered at the feature, provide the best performance in terms of both maximizing the distinctiveness and reducing the computational cost.

Once features have been extracted from map images, a set of all the candidate correspondences $\mathcal{C}$ between features in both images is determined by means of a measure of similarity between their descriptors that will be introduced in §14.4.2. Due to ambiguity in maps, it is common for a given feature to have several candidate correspondences. From all those candidates, a modified RANSAC algorithm obtains subsets of internally consistent hypotheses $\mathcal{C}_i \subset \mathcal{C}$ by imposing *uniqueness* (each feature must correspond up to just one in the other map) and the *rigid transformation* constraint (the relative position of features with respect to each other must be the same in both maps). The uncertainty of all the variables is accounted for during the whole process, thus all the decisions are taken upon stochastic tests. Unlike the standard RANSAC algorithm [Fis81], we propose to keep not only the solution with the largest number of supporting inliers but a dynamic number of them. Each of these detected hypotheses leads to a particular rigid transformation, which is modeled as a Gaussian distribution over the space of translations and rotations.

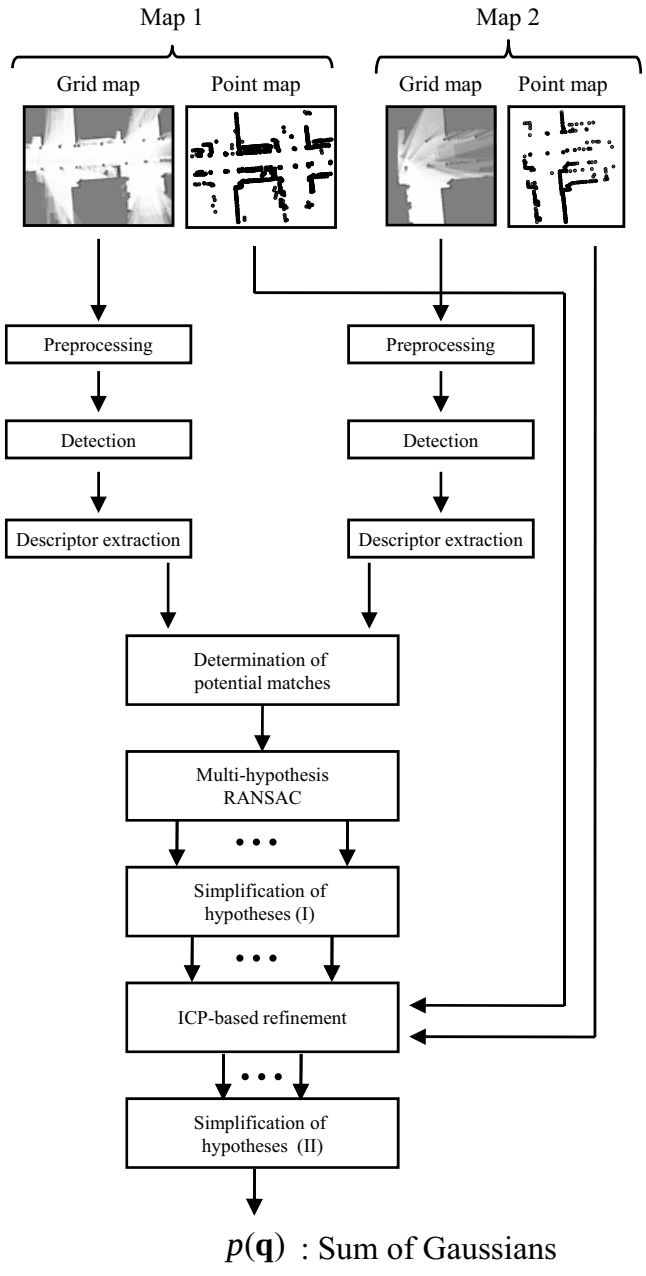


Figure 14.1: An overview of the proposed method for map matching, which aligns a pair of maps each comprising of a grid map and a point map. It firstly registers the grid maps to obtain a set of potential transformations $\mathbf{q}$, which are then refined employing the point maps and ICP-based alignment. The result is a probability density distribution for the actual $\mathbf{q}$ in the form of a mixture of Gaussians.

The general form for the probability distribution of the rigid transformation $\mathbf{q}$ between two maps, represented as a Sum of Gaussians (SOG), can be written down as:

$$p(\mathbf{q}) = \sum_i \mathcal{N}(\mathbf{q}; \mathbf{q}_i^\star, \mathbf{Q}_i)\omega_i \tag{14.2.1}$$

where each ω_i weights a Gaussian kernel centered at $\mathbf{q}_i^\star$ with covariance matrix $\mathbf{Q}_i$ and such that $\sum_i \omega_i = 1$. The distribution $p(\mathbf{q})$ can also be expanded using the law of total probability over all the potential sets of correspondences $\mathcal{C}_i$ as follows:

$$p(\mathbf{q}) = \sum_{\forall \mathcal{C}_i} p(\mathbf{q}|\mathcal{C}_i)P(\mathcal{C}_i) \tag{14.2.2}$$

Comparing Eq. (14.2.1) to Eq. (14.2.2) it is clear that we can choose $P(\mathcal{C}_i)$ as the SOG weights ω_i, and then model the density of $\mathbf{q}$ (given a set of correspondences $\mathcal{C}_i$) as a Gaussian distribution, that is:

$$p(\mathbf{q}|\mathcal{C}_i) = \mathcal{N}(\mathbf{q}; \mathbf{q}_i^\star, \mathbf{Q}_i) \tag{14.2.3}$$

The parameters of this distribution (its mean and covariance) will be derived in §14.7.1. A requirement for this probabilistic treatment is assigning a given uncertainty to the 2-d location of each detected feature. Following widely-accepted assumptions in the computer vision literature [Dav07, Sae06, Se01, Tam06], we take into account the following properties of the detected points:

- Since in most feature detectors each point is detected independently, Gaussian errors in the coordinates of different features are uncorrelated.
- As a consequence of this independent detection, all features may be assigned the same covariance.

- It is plausible for most interest point detectors to assume an isotropic distribution for the localization errors.

That is, the error in all features is modeled by the same Gaussian distribution and they are uncorrelated. This will be used in §14.7.1 to greatly simplify the derivation of the uncertainty in the optimal transformation.

The next sections address the issue of choosing an optimal combination of detector and descriptor, and then in §14.6 it will be recovered the problem of obtaining the subsets of correspondences $\mathcal{C}_i \in \mathcal{C}$ and the weights $\omega_i = P(\mathcal{C}_i)$.

14.3 Extraction of features

In this section we review some well-known image feature detectors and motivate the need for pre-processing the map images in order to improve the detection process.

14.3.1 Interest-point detectors

In a typical indoor occupancy grid map we can easily identify natural features produced by scene elements, like corners, columns or, in general, any sharp edge. They also appear in some outdoor maps originated by vertical poles, building corners, vehicle edges, etc. These natural landmarks are suitable for matching maps of the same areas since they naturally occur in the environment and they are typically static.

All those interest points can be detected by interpreting the grid map as a grayscale image, the map image, and applying existing key-point detectors. The most desirable property of any detector is its *repeatability*, that is, its ability to detect a given feature when it appears in different images.

We are interested in the performance of the following four methods:

- The Harris detector [Har88], which searches for points where the structure tensor has two large eigenvalues, revealing the existence of corners.

- The Kanade-Lucas-Tomasi (KLT) method ([Luc81, Shi94]) also relies on the structure tensor. It detects salient points where one of the eigenvalues exceeds a given threshold.

- The detection phase of the SIFT algorithm [Low99], which identifies scale-space extrema in pyramids of difference-of-Gaussians. This method aims at detecting *blobs* instead of corners [Mik02].

- The detector of SURF, based on an approximation to the Hessian matrix [Bay06].

There exists an issue in map images which affects the process of feature detection and needs to be handled appropriately. Grid mapping from laser range scans typically generates some artifacts in the maps which can be interpreted as high-frequency noise in the image (e.g. those arising from a single ray of the scans). To prevent the detection of spurious interest points in the middle of free-space, we propose to pre-process the images by applying first a Gaussian filter and then a median filter to attenuate most of the irregularities. Next we explain how we have tuned each filter for optimal detection performance.

14.3.2 Characterization

The dataset employed in this work consists of 10 pairs of grid maps created from real robot data. We must remark that the maps represent real loop-closure situations with partial overlap and small differences in the grids caused by noise and different viewpoints of the robot. Since hundreds of key points are detected in each of these

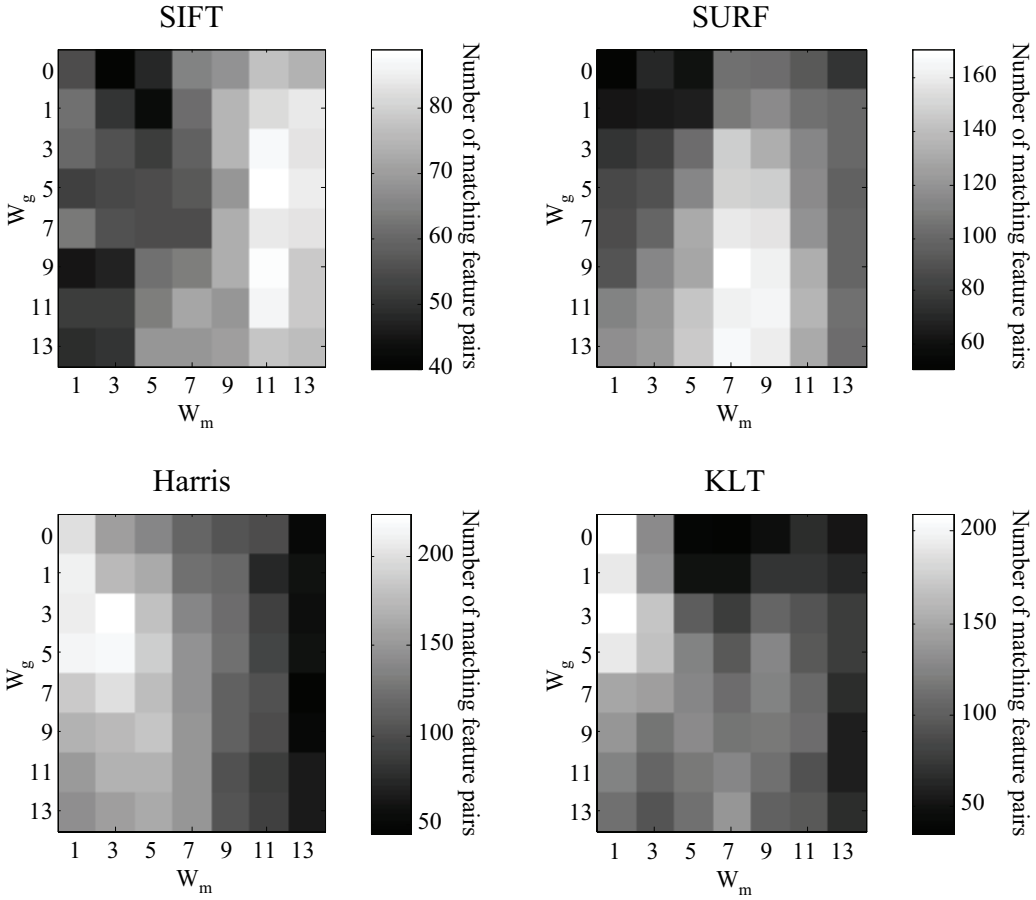


Figure 14.2: A measure of the repeatability for each detector and for different sizes of the Gaussian (W_g) and median (W_m) filters used to smooth the map images. Brighter colors indicate a higher number of common features detected in both maps.

grids, our overall characterization can be considered significant from a statistical point of view.

In order to evaluate the repeatability of each interest point detector we have applied it to both maps in each pair, and then counted the number of common detected features, i.e. the same feature must be detected in *both* grid maps. The correct pairings were obtained then from ground truth transformations between the pairs of maps, computed manually. To avoid a bias in our results due to the number of detected points, we have limited the number of interest points to a fixed value proportional to the extension of

each grid map (a typical value of 0.015 features per square meter is appropriate for all the maps employed in our comparison).

The results are summarized in Figure 14.2 for each detector and for different values of W_g and W_m, the sizes of the Gaussian and the median filter, respectively. The values $W_g = 0$ and $W_m = 1$ correspond to a null filter in each case, thus the cases of applying just one of the filters (or none of them) have been also accounted for.

Observe how blob detectors (SIFT and SURF) perform well for large filter sizes (that lead to more "softened" images), whereas corner detectors (Harris and KLT) have good repeatability for slightly filtered images or even for maps not filtered at all (refer to KLT results in Figure 14.2). Figure 14.3 shows an example of the different filters required by each detector to perform optimally. The best filter configuration for each detector has been employed in the benchmark presented in §14.5, and the corresponding overall number of matches can be seen in Figure 14.6(a).

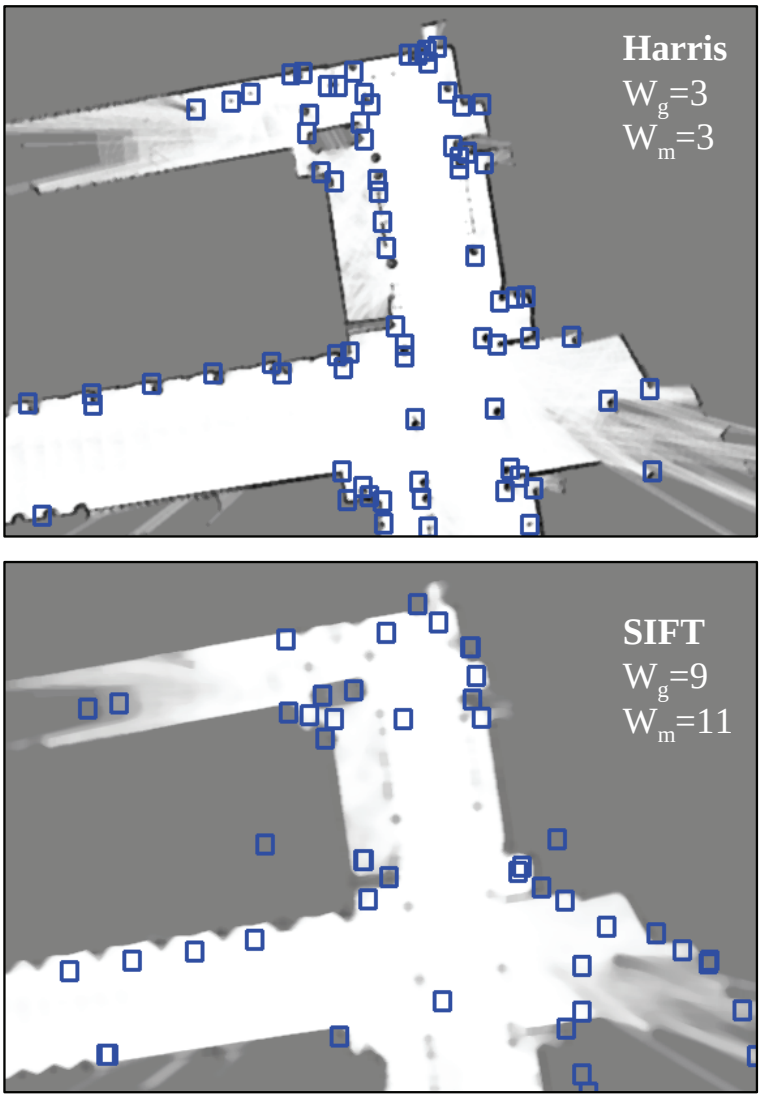


Figure 14.3: One of the maps from the dataset, filtered with a Gaussian and median filter of sizes W_g and W_m, respectively. Detected interest points are marked with small squares for the Harris and SIFT detectors. Notice how each method detects a different kind of features (corners or blobs), hence the different filtering requirements.

14.4 Descriptors

14.4.1 Review

Once the key-points are detected they are assigned distinctive descriptors in order to establish correspondences. We have studied the performance of the following five image descriptors [2] :

- **SIFT**: This method is based on histograms of image gradients [Low99], obtaining a 128-length descriptor vector.
- **SURF**: Based on the responses of Haar-wavelets as described in [Bay06].
- **Intensity-domain spin images (Spin)**: A 2D histogram of intensities indexed by distances [Laz03], with the maximum radius from the interest point determined by the parameter R_{max}. The usage of distances (disregarding angles), makes this descriptor rotation invariant.
- **Linear or logarithmic circular patches**: These two descriptors have many similarities, hence we discuss them here together. Both map a circular region of radius R_{max} centered at the interest point into a 2D matrix (the descriptor) of polar coordinates. Let this matrix be denoted by $\mathbf{f}(u, v)$, where the indices u and v stand for different values of the distance and the angle from the feature, respectively. The idea is to extract a circular patch of the neighborhood of the feature in a representation which is not invariant to rotations, but where these rotations become just shifts in the angle dimension (v), as illustrated with the examples in Figure 14.4(b)–(c). The only difference between the linear polar

[2] OpenCV implementations have been used for all the feature detectors and descriptors mentioned in this chapter, except for: (i) the SIFT method for which we rely on the implementation [Hes09] and (ii) the *lin-polar* descriptor, coded by the authors and submitted for publication in OpenCV 2.0.

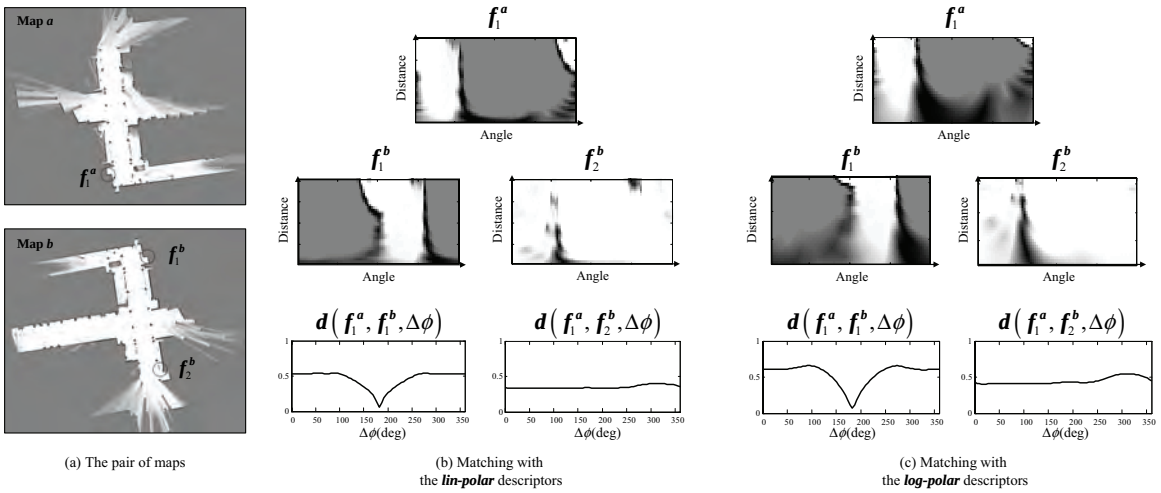


Figure 14.4: Example of matching features with different orientation. (a) An arbitrary reference feature f_1^a is highlighted in map a, and two potential pairings f_1^b (the real correspondence) and f_2^b are marked in map b. (b)–(c) The similarity between the feature descriptors is displayed as the distance function $d(f_i, f_j, \Delta\phi)$ for the cases of using the *lin-polar* and *log-polar* descriptors, respectively. Notice the pronounced minimum of the distance for the case of the real correspondence $f_1^a \leftrightarrow f_1^b$ close to the 180° relative rotation.

descriptor (*lin-polar* for short) and its logarithmic version (*log-polar*) is the usage of a linear or logarithmic scale in the distances.

Next we address the problem of measuring the similarity between descriptors, a requisite to evaluate their distinctiveness.

14.4.2 A similarity function between descriptors

Given a pair of descriptors $\mathbf{f}_i^a$ and $\mathbf{f}_j^b$ for two keypoints i and j from maps a and b, respectively, we are interested in measuring their similarity. For the SIFT, SURF and Spin descriptors the most natural measure is the Euclidean distance between the descriptor vectors. However, the cases of *lin-polar* and *log-polar* deserve more discussion since they are not directly invariant to orientation.

As illustrated in Figure 14.4, the descriptors of two matching features only differ by a shift in the angular dimension. Therefore, we propose to measure the distance

between two descriptors $\mathbf{f}_i$ and $\mathbf{f}_j$ by their Euclidean distance, given a rotation $\Delta\phi$, that is:

$$d\left(\mathbf{f}_i, \mathbf{f}_j, \Delta\phi\right) = \left(\sum_u \sum_v \left| \mathbf{f}_i(u,v) - \mathbf{f}_j(u, v+\Delta\phi) \right|^2 \right)^{\frac{1}{2}} \tag{14.4.1}$$

where the angular polar coordinate v is taken modulo the corresponding size of the matrix.

By computing the distance in Eq. (14.4.1) to a pair of descriptors $\mathbf{f}_i^a$ and $\mathbf{f}_j^b$ we obtain a distance vector for each possible shift in orientation $\Delta\phi$. As shown in Figure 14.4, these distance vectors have pronounced minimums for the true orientation when two features do really match, thus we propose to measure the inter-feature distance in the cases of *lin-polar* and *log-polar* as:

$$d\left(\mathbf{f}_i, \mathbf{f}_j\right) = \min_{\Delta\phi} d\left(\mathbf{f}_i, \mathbf{f}_j, \Delta\phi\right) \tag{14.4.2}$$

For all the descriptors in our comparison we have normalized distances to the range $[0, 1]$ in order to keep homogeneity in the results presented in the next section.

14.5 Benchmark of detectors and descriptors

14.5.1 Set up of the benchmark

After defining a similarity measure for pairs of descriptors in §14.4.2, we are interested in obtaining a set of *candidate correspondences* between the features of two maps a and b, given their descriptors $\mathbf{f}_i^a$ and $\mathbf{f}_j^b$. The goodness of all the potential correspondences must be evaluated such as only the most promising pairings (those passing a given

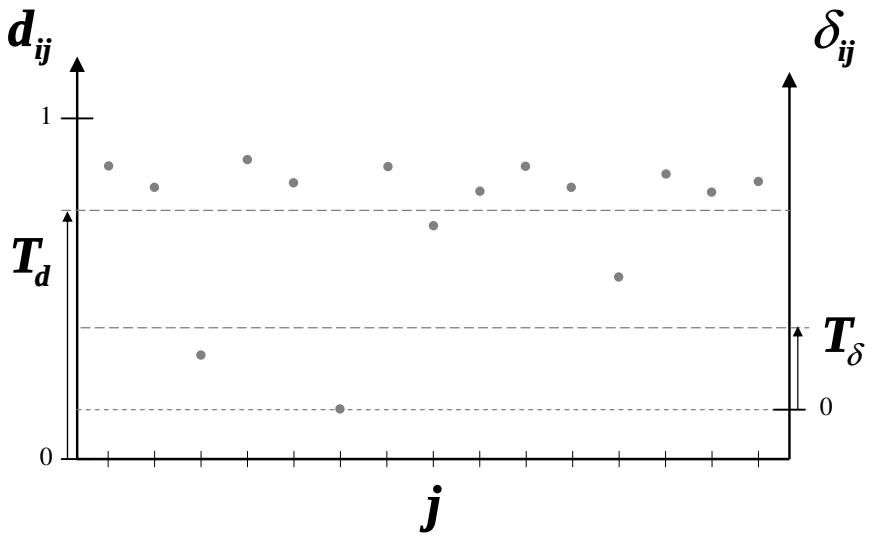


Figure 14.5: A schematic illustration of the distance between descriptors d_{ij} and the index δ_{ij}, which measures those distances relative to the closest one for each given feature i. Note that, by definition, the best pairing is always assigned a value $\delta_{ij} = 0$. A pairing will be accepted only if it is below both thresholds T_d (absolute) and T_δ (relative to the minimum distance).

test) are considered as candidates. It is acceptable for each feature to have multiple potential correspondences in the other map, since a subsequent robust matching step (such as RANSAC [Fis81]) can easily manage that ambiguity.

The arguably simplest test for selecting matchings is thresholding, which in our case means to accept a potential match between $\mathbf{f}_i^a$ and $\mathbf{f}_j^b$ only if the distance d_{ij} between their descriptors is below a fixed value T_d. However, this simple scheme has some drawbacks in the context of grid matching, because distance values between actually corresponding pairs may vary in a relatively large range. Thus, any permissive threshold T_d which covers most of the good correspondences would suffer from a high rate of false positives.

Following an idea similar to Lowe's proposal in [Low99] we introduce a second condition for establishing candidate pairings: the associated distance d_{ij} must be not

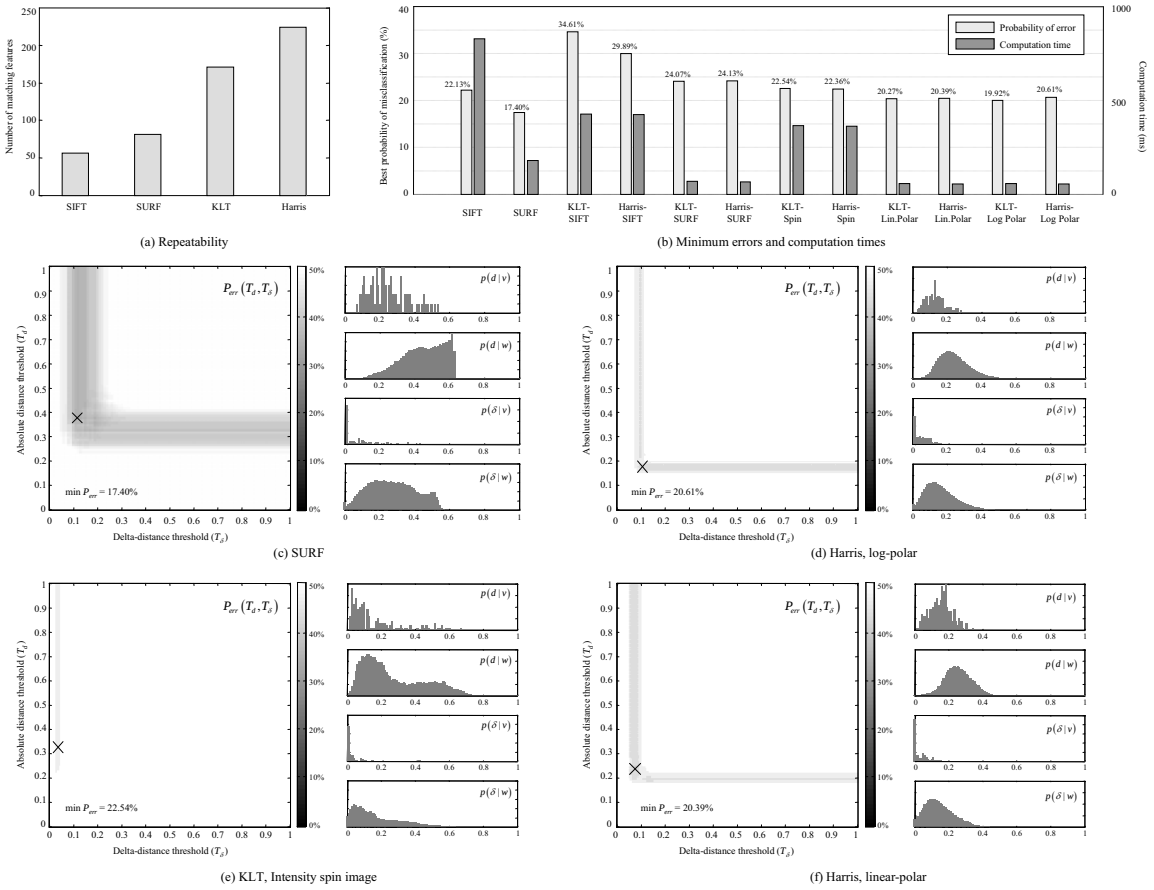


Figure 14.6: The benchmark of feature detectors for grid map matching. (a) A measure of the repeatability for each detector. (b) For each combination of detector and descriptor, the resulting overall probability of classification error P_{err} for its best thresholds, i.e. that marked with a cross in (c)–(f), along with its average computation time for one map. (c)–(f) Four examples of the expected P_{err} for different values of thresholds T_d and T_δ. The point with the minimum P_{err} is marked with a cross in each figure. We have also shown the marginal conditional distributions for the distance d and the distance-difference δ for valid (v) and wrong (w) associations are shown on the right hand of each subfigure.

only below the threshold T_d, but also sufficiently close to the best matching of $\mathbf{f}_i^a$ in map b, that is, the minimum of d_{ij} for all values of j (see Figure 14.5). This restriction is characterized by a second threshold T_δ which states the maximum acceptable distance δ between a potential pairing and the best one, that is, $\delta_{ij} = d_{ij} - \min_j d_{ij}$. Notice that for the extreme case $T_\delta = 0$ each feature will be associated to only one in the other map: the one with the closest descriptor. Both measures d_{ij} and δ_{ij} are illustrated with an example in Figure 14.5 for clarity.

A benchmark has been carried out to obtain the optimal values for the thresholds T_d and T_δ from a training set of 10 pairs of submaps with known ground-truth and for several combinations of detectors and descriptors. Optimal thresholds have been determined by minimizing the probability P_{err} of misclassifying a correspondence as a valid or an invalid candidate, given by:

$$\begin{aligned} P_{err}(T_d, T_\delta) &= P(w)P_{err}(T_d, T_\delta|w) + P(v)P_{err}(T_d, T_\delta|v) \\ &= P(w)P(d_{ij} < T_d, \delta_{ij} < T_\delta|w) \\ &+ P(v)\left[1 - P(d_{ij} < T_d, \delta_{ij} < T_\delta|v)\right] \end{aligned} \tag{14.5.1}$$

which can be evaluated given knowledge of the joint densities $p(d, \delta|v)$ and $p(d, \delta|w)$, where v and w stand for valid and wrong pairings, respectively. The expression above can be easily derived by noticing that a misclassification will occurs when: (i) a distance d_{ij} passes both thresholds and it was a wrong association (first term in the sum), or (ii) a valid pairing does not pass the thresholds (second term). For our analysis we assume no a priori information about the probability of being in a valid or invalid pairing, thus we have $P(v) = P(w) = 1/2$. The joint conditional densities $p(d_{ij}, \delta_{ij}|v)$ and $p(d_{ij}, \delta_{ij}|w)$ have been estimated from histograms generated by evaluating all the potential pairings in the 10 pairs of submaps, which amounts to 220 valid and 240,000

invalid correspondences.

The results of the benchmark are summarized in Figure 14.6(e) which shows the minimum classification error P_{err} attainable by each combination of feature detector and descriptor, along the associated average computation time (for one whole submap). These times include detection, descriptor extraction and distance computations, but they do not include the preprocessing filters discussed in §14.3.2. This preprocessing would add an average of 10 to 200ms, with larger computational burdens associated to SIFT and SURF since they require larger filter kernels than the Harris or KLT methods.

Please, notice that for those descriptors parameterized by a maximum radius R_{max} (see §14.4.1) we present the results only for the value that minimizes the classification error. However, this is a non-critical parameter since any value in the range $1 - 3$ meters gives similar results.

14.5.2 Results of the benchmark

The first important conclusion we can extract from our comparison is that no descriptor can tell valid pairings from wrong ones with a classification error below $\sim 20\%$, which is clearly a consequence of the ambiguity of features in map images where many ones look quite similar locally. Still, discarding $\sim 80\%$ of the wrong pairings provides an invaluable improvement to the subsequent robust matching algorithm, since it will have to deal with a reduced fraction of outliers.

It is interesting to note that the SIFT and SURF descriptors have a much poorer performance when computed for interest points localized by the Harris or the KLT detectors (third to sixth values in the bar graph) than when computed as proposed in their original methods (the first two values in the graph). As commented in §14.3.1

and illustrated in Figure 14.6(f), this has important consequences for the practical applicability of those descriptors to grid matching, since the original SIFT and SURF detectors have poorer repeatability than the Harris and KLT methods. Subsequently, we discard the usage of these two descriptors as the optimal solution since they lead to quite similar error ratios (P_{err}) than the other descriptors while severely reducing the number of matched points and implying a higher computational burden, as can be seen in Figure 14.6(e).

In Figure 14.6(a)–(d) it is represented the computed $P_{err}(T_d, T_\delta)$ for some selected methods along the marginal distributions obtained in our benchmark. Observe how the marginal $p(\delta_{ij}|v)$ presents a clear peak at the origin ($\delta_{ij} = 0$) for all the methods, which indicates that the closest feature is often the actual correspondence [3]. However, this is not always the case, hence the optimal T_δ values are not exactly zero.

Notice that the worst obtained value for P_{err} (0.5, represented in white in the graphs) is obtained for a wide range of threshold values, while more reduced error ratios only appear for a certain band of the parameters (represented by darker areas). The thickness of these bands is related to the distinctiveness of the descriptors, as can be observed in the densities of descriptor distances for valid and wrong pairings (the histograms at the right hand of each P_{err} graph). For instance, compare the histograms $p(d|v)$ and $p(d|w)$ for the SURF and the Spin descriptors in Figure 14.6(a)-(c), where it is clear that in SURF the histograms concentrate in relatively different areas (easing the decision of where to place the threshold) whereas this is definitively not the case for the Spin descriptor.

As a final conclusion from our benchmark, the *lin-polar* and *log-polar* descriptors, both with virtually identical performance, emerge as the best choices for grid matching

[3] Recall that, by definition, $\delta_{ij} = 0$ means that feature $\mathbf{f}_j$ has the minimum distance to feature $\mathbf{f}_i$.

in combination with either Harris or KLT detector, due to their reduced misclassification probability and faster computation time.

14.6 Construction of the map transformation SOG

14.6.1 The modified RANSAC algorithm

Subsets of self-consistent correspondences $\mathcal{C}_i \in \mathcal{C}$ can be extracted with RANSAC, a consensus-based method able of telling inliers from outliers [Fis81]. However, in our problem it is not enough to keep the hypothesis with most supporting inliers since ambiguity in grid matching can lead to multiple, mutually incompatible but internally consistent subsets $\mathcal{C}_i$. We propose instead to maintain each of those hypotheses as a Gaussian mode in the SOG (refer to Eq. (14.2.2)), hence the need to modify the RANSAC algorithm to allow the existence of multiple hypotheses.

Next we describe the complete process, which has been stated in Algorithm 4 for clarity. Firstly, two "seed" correspondences (the minimum number required to unequivocally determine the distribution of the associated map transformation $p(\mathbf{q}|\mathcal{C}_i)$) are randomly chosen from $\mathcal{C}$ to initialize the subset $\mathcal{C}_i = \{c_{k_1}, c_{k_2}\}$. The *uniqueness* constraint is tested first, that is, in a valid pairing one given feature cannot appear in both correspondences c_{k_1} and c_{k_2} simultaneously. Then, the feasibility of this pair is tested by a chi-square test which detects inconsistencies between the inter-feature spatial distances d_a and d_b measured in the two maps a and b (refer to the example in Figure 14.7(a)). As shown in the Appendix E, if

$$\frac{(d_a^2 - d_b^2)^2}{8\sigma^2(d_a^2 + d_b^2)} < \chi^2_{1,c} \tag{14.6.1}$$

Algorithm 4 robust_transform $(\mathcal{C}, \{p_i^A\}, \{p_j^B\}) \rightarrow SOG$

$SOG \Leftarrow \varnothing$
$iter \Leftarrow 1$
repeat // *RANSAC iterations*
  $\hat{\mathcal{C}} \Leftarrow \{c_{k_1}, c_{k_2}\} \subset \mathcal{C} / uniqueness(c_{k_1}, c_{k_2})$
  if $D_M^2(c_{k_1}, c_{k_2}) < \chi_{c,1}^2$ **then** // *Consistency test*
    if $\exists k / \hat{\mathcal{C}} \subset SOG_k.\hat{\mathcal{C}}$ **then** // *Already?*
      // *Increment the weight*
      $SOG_k.\omega \Leftarrow SOG_k.\omega + 1$
    else
      // *It is a new SOG mode*
      repeat // *Incorporate inliers*
        $(q_i^\star, \mathbf{Q}_i^\star) \Leftarrow opt\_transf(\hat{\mathcal{C}})$ // *See Eqs.(14.7.3),(14.7.17)*
        $(i^\star, j^\star) \Leftarrow \arg\max_{(i,j)} \int p_i(\xi)\tilde{p}_j(\xi)d\xi$
        if $D_M^2(i^\star, j^\star) < \chi_{c,2}^2$ **then**
          $\hat{\mathcal{C}} \Leftarrow \hat{\mathcal{C}} \cup (i^\star, j^\star)$ // *Accept pairing*
        end if
      until $D_M^2(i^\star, j^\star) \geq \chi_{c,2}^2$
      if $|\hat{\mathcal{C}}| \geq M$ **then** // *Minimum inlier support*
        // *New Gaussian mode with* $\omega = 1$
        $SOG \Leftarrow SOG \cup (\hat{\mathcal{C}}, 1, (q_i^\star, \mathbf{Q}_i))$
      end if
    end if
  end if
  $iter \Leftarrow iter + 1$
until $iter > maxIters$

holds, we can accept that the distances are consistent within a confidence of c, where $\chi_{n,c}^2$ stands for the inverse chi-square cumulative distribution with n degrees of freedom.

Next, it must be determined the number of inliers supporting the hypothesis $p(q|\mathcal{C}_i)$ defined by each set of initial pairings $\mathcal{C}_i$. This is achieved by establishing associations between all the features in map b and those in a transformed by $\mathbf{q}$. Notice that this is a stochastic data-association problem since all feature locations, and the transformation itself, have associated uncertainties.

A robust method for stochastic data association is the Joint Compatibility Branch

and Bound (JCBB) [Nei01], but unfortunately its exponential time complexity makes it impractical for our problem, where each map will typically contain about one hundred features.

Our alternative, which has been included in Algorithm 4, consists of sequentially incorporating matches which optimize the integral of the product of the two Gaussians, which can be shown to reflect the confidence of a potential pairing. The incorporation of inliers stops when the next best pairing candidate (i, j) has a squared Mahalanobis distance $D_M^2(i, j)$ above a given threshold $\chi_{2,c}^2$.

The above process is repeated a number of times, updated dynamically as new inliers are found [Har03]. Regarding the weights of the SOG, each Gaussian mode is initially assigned a unit weight, which is incremented each time the same subset of correspondences is found in subsequent iterations (this implies that, in the end, weights must be normalized). An enhancement of this approach is to test whether the two first correspondences $\mathcal{C}_i$ are already part of another $\mathcal{C}_j$, and in that case, to increment the weight ω_j. This heuristic is justified by the observation that the same set of self-consistent pairings will be likely obtained if any pair of them is selected as the two first "seed" correspondences.

Notice as well the existence of a minimum number of required inliers (pairings) M in order to accept a hypothesis. In our experiments, this threshold has been heuristically set to a $\sim 15\%$ of the average number of features found in each map. This restriction prevents the detection of spurious hypotheses with very few supporting inliers caused by pure chance when two maps do not really match.

14.6.2 Refinement and merge of Gaussian modes

As depicted in Figure 14.1, we propose to *simplify* the SOG generated by the RANSAC stage before using the point maps to refine it further on. This means that, whenever possible, two or more Gaussians are replaced by just one with the appropriate mean and covariance such that it closely covers the same volume than the original pair. One of the reasons to simplify the SOG is to reduce as much as possible the cost of the following *refinement* step, in which ICP [Bes92] is applied to the point maps in order to improve the estimate of the mean map transformation $q^\star$. The resulting SOG is then tested again for further potential simplifications as illustrated in Figure 14.1, obtaining the final, possibly multi-modal, density distribution for the map transformation.

We follow the method proposed in [Run07] to measure the Kullback-Leibler divergence (KLD) (see §2.6) between the original and tentative densities, and only those simplifications with a KLD value below a threshold are admitted.

14.7 The optimal solution to 2-d matching and its uncertainty

Given a set of point correspondences from two different frames of reference a and b, it is well-known that a closed-form solution exists for finding the 2-d rigid transformation between them that is optimal in the sense of least mean square error (LMSE) [Lu97b, Mar06]. In mobile robotics, this solution is best known for its role within the Iterative Closest Point (ICP) algorithm [Bes92], widely employed for aligning pairs of laser range scans [Lu97a, Lu97b].

This section reviews that optimal solution and derives an estimation of its uncertainty. Taking the uncertainty into account is essential, since the position of the points

involved in the correspondences are always prone to error (e.g. sensor noise). The resulting expression is proven to be accurate and realistic by means of a comparison to Monte Carlo simulations.

The derivation will focus on the specific case of two 2-d maps a and b, where the following assumptions are made about the *feature points* paired in the set of correspondences: (i) Error in feature points can be modeled as an isotropic 2-d Gaussian, with characteristic variance σ_p^2, and (ii) all the errors are independent, i.e. there is a null covariance between different features. These assumptions will allow important simplifications in our derivation, as will be seen in the next section.

14.7.1 Uncertainty of the optimal transformation

Given a certain set of feature correspondences $\mathcal{C}_i$, we model the probability density of a rigid transformation $\mathbf{q} = [x\ y\ \phi]^\top$ between the two frames of reference as a Gaussian distribution, that is:

$$p(\mathbf{q}|\mathcal{C}_i) = \mathcal{N}(\mathbf{q}; \mathbf{q}_i^\star, \mathbf{Q}_i) \tag{14.7.1}$$

where $\mathbf{q}_i^\star$ and $\mathbf{Q}_i$ represent the corresponding mean and covariance matrix, respectively. In the following we derive expressions for the parameters of this distribution. Note that the subscript i is a constant throughout the whole derivation, but it is still convenient since it reminds us that the derived Gaussian represents the pdf for just one (the i'th) hypothesis within the SOG.

The mean of the Gaussian will be given by the optimal solution $\mathbf{q}_i^\star$, while the covariance matrix is approximated by first-order linearization.

Let $E_{\mathcal{C}_i}(\mathbf{q})$ be the squared error in the matching of features from maps a and b, for a rigid transformation $\mathbf{q}$ and a given set of correspondences $\mathcal{C}_i$, defined as:

$$E_{\mathcal{C}_i}(\mathbf{q}) = \sum_{\forall c_k \in \mathcal{C}_i} \left| \mathbf{p}_i^b - (\mathbf{q} \oplus \mathbf{p}_j^a) \right|^2 \tag{14.7.2}$$

where $\oplus$ represents the pose composition operator (see Appendix A) and $\mathbf{p}_k^m$ stands for the position of the k'th feature in the frame of reference m. For the 2-d case, in which we are interested, the optimal transformation $\mathbf{q}_i^\star$ that minimizes this error can be obtained by equaling the derivative of Eq. (14.7.2) to zero, which leads to the closed-form solution [Lu97b]:

$$\mathbf{q}_i^\star = \begin{bmatrix} x_i^\star \\ y_i^\star \\ \phi_i^\star \end{bmatrix} = \begin{bmatrix} \bar{x}_a - \bar{x}_b \cos\phi_i^\star + \bar{y}_b \sin\phi_i^\star \\ \bar{y}_a - \bar{x}_b \sin\phi_i^\star - \bar{y}_b \cos\phi_i^\star \\ \tan^{-1}\left(\frac{\Delta_y}{\Delta_x}\right) \end{bmatrix} \tag{14.7.3}$$

where $\bar{x}_a$, $\bar{y}_a$, $\bar{x}_b$, and $\bar{y}_b$ are the mean values of the vectors $\mathbf{x}_a$, $\mathbf{y}_a$, $\mathbf{x}_b$, and $\mathbf{y}_b$, respectively, which contain the 2-d coordinates of features within the maps. We have also employed the terms Δ_x and Δ_y, defined as:

$$\begin{aligned} \Delta_x &= N\left(\sum_k x_k^a x_k^b + \sum_k y_k^a y_k^b\right) - N^2\left(\bar{x}_a\bar{x}_b + \bar{y}_a\bar{y}_b\right) \\ \Delta_y &= N\left(\sum_k y_k^a x_k^b - \sum_k x_k^a y_k^b\right) + N^2\left(\bar{x}_a\bar{y}_b - \bar{y}_a\bar{x}_b\right) \end{aligned} \tag{14.7.4}$$

with $N = |\mathcal{C}_i|$ standing for the number of pairings in $\mathcal{C}_i$.

Before proceeding with the derivation, it is more convenient to rewrite Eq. (14.7.3) in an alternative form to avoid the dependency of the first two components $(x_i^\star, y_i^\star)$ on the third one $(\phi_i^\star)$. This can be easily done by applying basic trigonometric definitions, obtaining:

$$\mathbf{q}_i^\star = \begin{bmatrix} \bar{x}_a - \bar{x}_b \frac{\Delta_x}{\sqrt{\Delta_x^2+\Delta_y^2}} + \bar{y}_b \frac{\Delta_y}{\sqrt{\Delta_x^2+\Delta_y^2}} \\ \bar{y}_a - \bar{x}_b \frac{\Delta_y}{\sqrt{\Delta_x^2+\Delta_y^2}} - \bar{y}_b \frac{\Delta_x}{\sqrt{\Delta_x^2+\Delta_y^2}} \\ \tan^{-1}\left(\frac{\Delta_y}{\Delta_x}\right) \end{bmatrix} \tag{14.7.5}$$

The optimal transformation in Eq. (14.7.5) is therefore a function $\mathbf{q}(\cdot)$ of six auxiliary variables $\mathbf{z} = [\bar{x}_a\ \bar{y}_a\ \bar{x}_b\ \bar{y}_b\ \Delta_x\ \Delta_y]^\top$, that is, $\mathbf{q}_i^\star = \mathbf{q}(\mathbf{z})$. In order to estimate the covariance matrix $\mathbf{Q}_i$ that models its uncertainty, it is firstly needed the multivariate Gaussian distribution of the vector of auxiliary variables $\mathbf{z}$. This vector is a function of the 2-d coordinates of all the features $\mathbf{x}_a$, $\mathbf{y}_a$, $\mathbf{x}_b$ and $\mathbf{y}_b$. Assuming that these coordinates are corrupted with an additive, zero-mean Gaussian noise with known covariance matrices $\mathbf{X}_a$, $\mathbf{Y}_a$, $\mathbf{X}_b$ and $\mathbf{Y}_b$, we can obtain the covariance of $\mathbf{z}$ from:

$$\boldsymbol{\Sigma}_\mathbf{z} \approx \mathbf{J}_z \begin{bmatrix} \mathbf{X}_a & 0 & 0 & 0 \\ 0 & \mathbf{Y}_a & 0 & 0 \\ 0 & 0 & \mathbf{X}_b & 0 \\ 0 & 0 & 0 & \mathbf{Y}_b \end{bmatrix} \mathbf{J}_z^\top \tag{14.7.6}$$

Since $\mathbf{z}$ depends on the whole set of feature coordinates, the Jacobian matrix $\mathbf{J}_z$ has a dimensionality of $6 \times 4N$. In despite of the large size of the matrices involved in Eq. (14.7.6), important simplifications are possible because of our assumptions of independence and isotropic distributions (§14.7). Therefore, the covariances $\mathbf{X}_a$, $\mathbf{Y}_a$, $\mathbf{X}_b$ and $\mathbf{Y}_b$ are diagonal matrices with the same variance σ_p^2 for all the coordinates, that is:

$$\mathbf{X}_a = \mathbf{Y}_a = \mathbf{X}_b = \mathbf{Y}_b = \sigma_p^2 \mathbf{I}_N \tag{14.7.7}$$

Therefore, the covariance in Eq. (14.7.6) becomes:

$$\mathbf{\Sigma_z} \;=\; \sigma_p^2 \mathbf{J}_z \mathbf{J}_z^\top \tag{14.7.8}$$

The Jacobian matrix $\mathbf{J}_z$ can be easily obtained from the derivatives of the six components of $\mathbf{z} = [\bar{x}_a\ \bar{y}_a\ \bar{x}_b\ \bar{y}_b\ \Delta_x\ \Delta_y]^\top$ with respect to the sequence of coordinates x_a, y_a, x_b and y_b:

$$\mathbf{J}_z = \begin{bmatrix} \frac{1}{N} & \cdots & \frac{1}{N} & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & \cdots & 0 & \frac{1}{N} & \cdots & \frac{1}{N} & 0 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & \cdots & 0 & 0 & \cdots & 0 & \frac{1}{N} & \cdots & \frac{1}{N} & 0 & \cdots & 0 \\ 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & \frac{1}{N} & \cdots & \frac{1}{N} \\ \cdots & \frac{\partial \Delta_x}{\partial x_a^k} & \cdots & \cdots & \frac{\partial \Delta_x}{\partial y_a^k} & \cdots & \cdots & \frac{\partial \Delta_x}{\partial x_b^k} & \cdots & \cdots & \frac{\partial \Delta_x}{\partial y_b^k} & \cdots \\ \cdots & \frac{\partial \Delta_y}{\partial x_a^k} & \cdots & \cdots & \frac{\partial \Delta_y}{\partial y_a^k} & \cdots & \cdots & \frac{\partial \Delta_y}{\partial x_b^k} & \cdots & \cdots & \frac{\partial \Delta_y}{\partial y_b^k} & \cdots \end{bmatrix}_{6\times 4N} \tag{14.7.9}$$

Now, the partial derivatives of Δ_x and Δ_y in the above matrix can be computed from their expressions in Eq. (14.7.4), leading to:

$$\begin{array}{rclrcl} \frac{\partial \Delta_x}{\partial x_a^k} &=& N x_b^k - \sum_k x_b^k & \frac{\partial \Delta_y}{\partial x_a^k} &=& \sum_k y_b^k - N y_b^k \\ \frac{\partial \Delta_x}{\partial y_a^k} &=& N y_b^k - \sum_k y_b^k & \frac{\partial \Delta_y}{\partial y_a^k} &=& N x_b^k - \sum_k x_b^k \\ \frac{\partial \Delta_x}{\partial x_b^k} &=& N x_a^k - \sum_k x_a^k & \frac{\partial \Delta_y}{\partial x_b^k} &=& N y_a^k - \sum_k y_a^k \\ \frac{\partial \Delta_x}{\partial y_b^k} &=& N y_a^k - \sum_k y_a^k & \frac{\partial \Delta_y}{\partial y_b^k} &=& \sum_k x_a^k - N x_a^k \end{array} \tag{14.7.10}$$

In order to derive an expression for the terms of $\mathbf{\Sigma_z}$ in Eq. (14.7.8), let $H(i,j)$ denote the elements in the matrix $\mathbf{J}_z\mathbf{J}_z^\top$. Thus, we have:

$$H(i,j) = \sum_{k=1}^{4N} \mathbf{J}_z(i,k)\mathbf{J}_z(j,k) \tag{14.7.11}$$

that is, the "dot product" of the i'th and j'th rows of the Jacobian. Since the first four rows of $\mathbf{J}_z$ are orthogonal (see Eq. (14.7.9)), it comes out that the first 4×4 submatrix

of $\mathbf{J}_z\mathbf{J}_z^\top$ is diagonal, with its non-zero elements given by $H(i,i) = \frac{N}{N^2} = \frac{1}{N}$. The rest of elements are not trivial to compute, thus we will assign them names in order to derive them next:

$$\mathbf{\Sigma_z} = \left[\begin{array}{cccc|cc} & & & & \sigma_{\bar{x}_a\Delta_x} & \sigma_{\bar{x}_a\Delta_y} \\ & & \frac{\sigma_p^2}{N}\mathbf{I_4} & & \sigma_{\bar{y}_a\Delta_x} & \sigma_{\bar{y}_a\Delta_y} \\ & & & & \sigma_{\bar{x}_b\Delta_x} & \sigma_{\bar{x}_b\Delta_y} \\ & & & & \sigma_{\bar{y}_b\Delta_x} & \sigma_{\bar{y}_b\Delta_y} \\ \hline \sigma_{\bar{x}_a\Delta_x} & \sigma_{\bar{y}_a\Delta_x} & \sigma_{\bar{x}_b\Delta_x} & \sigma_{\bar{y}_b\Delta_x} & \sigma^2_{\Delta_x} & \sigma_{\Delta_x\Delta_y} \\ \sigma_{\bar{x}_a\Delta_y} & \sigma_{\bar{y}_a\Delta_y} & \sigma_{\bar{x}_b\Delta_y} & \sigma_{\bar{y}_b\Delta_y} & \sigma_{\Delta_x\Delta_y} & \sigma^2_{\Delta_y} \end{array}\right] \tag{14.7.12}$$

If we start evaluating, for example, the cross-covariance of $\bar{x}_a$ and Δ_x, it must be evaluated the "dot product" of the first and fifth rows of Eq. (14.7.9):

$$\begin{aligned} \frac{\sigma_{\bar{x}_a\Delta_x}}{\sigma_p^2} &= \frac{1}{N}\sum_k \frac{\partial\Delta_x}{\partial x_a^k} = \frac{1}{N}\sum_k \left(N x_b^k - \sum_k x_b^k\right) \\ &= \sum_k \left(x_b^k - \bar{x}_b\right) = N \cdot E\left[\mathbf{x}_b - \bar{x}_b\right] = 0 \end{aligned} \tag{14.7.13}$$

Observe how the results is *exactly* zero due to the definition of the average value $\bar{x}_b$. It can be verified that all off-diagonal entries in Eq. (14.7.12) also amount to zero, thus $\mathbf{\Sigma_z}$ is a diagonal matrix: all the auxiliary variables in $\mathbf{z}$ are uncorrelated.

Regarding the two diagonal elements $\sigma^2_{\Delta_x}$ and $\sigma^2_{\Delta_y}$, it can be proven that both are equal to $\beta\sigma_p^2$, with:

$$\beta = N^2(N-1)\left(\hat{\sigma}^2_{x_a} + \hat{\sigma}^2_{y_a} + \hat{\sigma}^2_{x_b} + \hat{\sigma}^2_{y_b}\right) \tag{14.7.14}$$

where the constants $\hat{\sigma}^2_{x_a}$, $\hat{\sigma}^2_{y_a}$, $\hat{\sigma}^2_{x_b}$ and $\hat{\sigma}^2_{y_b}$ represent the unbiased estimates of the variance for their corresponding vectors. Therefore, it has been obtained an expression for the covariance of $\mathbf{z}$:

$$\mathbf{\Sigma_z} = \sigma_p^2 diag\left(\frac{1}{N}\ \frac{1}{N}\ \frac{1}{N}\ \frac{1}{N}\ \beta\ \beta\right) \tag{14.7.15}$$

At this point we can proceed with the derivation of the covariance of $\mathbf{q}_i^\star$. We compute now the Jacobian $\mathbf{J}_q$ of Eq. (14.7.5) with respect to $\mathbf{z}$, which can easily be shown to be:

$$\mathbf{J}_q = \begin{bmatrix} 1 & 0 & -\frac{\Delta_x}{\sqrt{\Delta_x^2+\Delta_y^2}} & \frac{\Delta_y}{\sqrt{\Delta_x^2+\Delta_y^2}} & -\frac{\bar{x}_b\Delta_y^2+\bar{y}_b\Delta_x\Delta_y}{\left(\Delta_x^2+\Delta_y^2\right)^{3/2}} & \frac{\bar{x}_b\Delta_x\Delta_y+\bar{y}_b\Delta_x^2}{\left(\Delta_x^2+\Delta_y^2\right)^{3/2}} \\ 0 & 1 & -\frac{\Delta_y}{\sqrt{\Delta_x^2+\Delta_y^2}} & -\frac{\Delta_x}{\sqrt{\Delta_x^2+\Delta_y^2}} & \frac{\bar{x}_b\Delta_x\Delta_y-\bar{y}_b\Delta_y^2}{\left(\Delta_x^2+\Delta_y^2\right)^{3/2}} & \frac{-\bar{x}_b\Delta_x^2+\bar{y}_b\Delta_x\Delta_y}{\left(\Delta_x^2+\Delta_y^2\right)^{3/2}} \\ 0 & 0 & 0 & 0 & -\frac{\Delta_y}{\Delta_x^2+\Delta_y^2} & \frac{\Delta_x}{\Delta_x^2+\Delta_y^2} \end{bmatrix} \tag{14.7.16}$$

This Jacobian can finally be employed to propagate the uncertainty from $\mathbf{z}$ to our optimal estimate $\mathbf{q}_i$ by means of:

$$\mathbf{Q}_i = \mathbf{J}_q\mathbf{\Sigma}_z\mathbf{J}_q^\top = \sigma_p^2\begin{pmatrix} C_{11} & C_{12} & C_{13} \\ C_{12} & C_{22} & C_{23} \\ C_{13} & C_{23} & C_{33} \end{pmatrix} \tag{14.7.17}$$

where, after replacing the values of Eq. (14.7.15)-(14.7.16) and operating, it is obtained:

$$\begin{aligned} C_{11} &= \frac{2}{N} + \beta\left(\frac{\bar{x}_b\Delta_y + \bar{y}_b\Delta_x}{\Delta_x^2+\Delta_y^2}\right)^2 \\ C_{22} &= \frac{2}{N} + \beta\left(\frac{\bar{x}_b\Delta_x - \bar{y}_b\Delta_y}{\Delta_x^2+\Delta_y^2}\right)^2 \\ C_{33} &= \frac{\beta}{\Delta_x^2+\Delta_y^2} \\ C_{12} &= \beta\frac{\left(\bar{x}_b\Delta_y + \bar{y}_b\Delta_x\right)\left(\bar{y}_b\Delta_y - \bar{x}_b\Delta_x\right)}{\left(\Delta_x^2+\Delta_y^2\right)^2} \\ C_{13} &= \beta\frac{\bar{x}_b\Delta_y + \bar{y}_b\Delta_x}{\left(\Delta_x^2+\Delta_y^2\right)^{\frac{3}{2}}} \\ C_{23} &= \beta\frac{\bar{y}_b\Delta_y - \bar{x}_b\Delta_x}{\left(\Delta_x^2+\Delta_y^2\right)^{\frac{3}{2}}} \end{aligned} \tag{14.7.18}$$

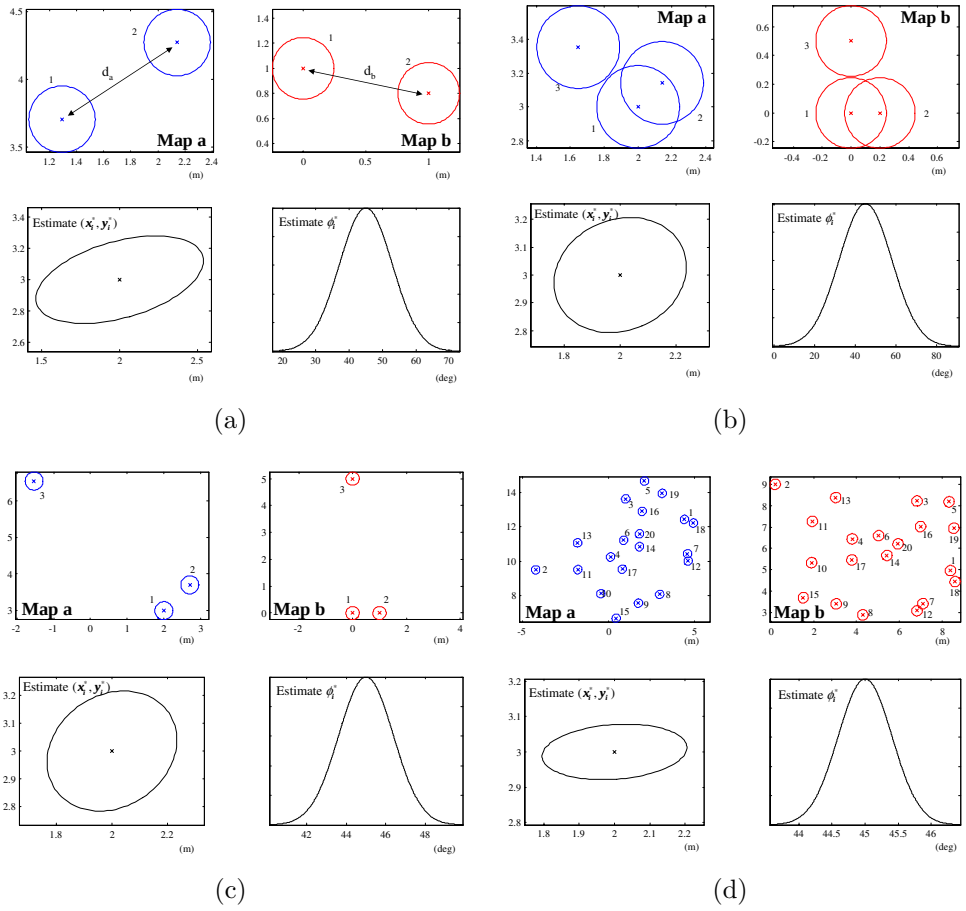


Figure 14.7: Four sets of correspondences between two synthetic maps a and b for different spatial distributions and number of detected features. Here, the position uncertainty for all the features has been set to $\sigma_p = 0.10$ and ellipses represent 95% confidence intervals. The inter-feature distances measured in the different maps, d_a and d_b, as employed in the Appendix E, are shown in (a) as an example.

To summarize this section, the optimal transformation $\mathbf{q}_i$ has been modeled as a Gaussian distribution with mean $\mathbf{q}_i^\star$ given by Eq. (14.7.3) and covariance $\mathbf{Q}_i$ by Eq. (14.7.17)–(14.7.18).

It is important to highlight that this covariance matrix can be computed in $O(N)$ operations thanks to all the simplifications exploited along its derivation, whereas the naïve implementation leads to a complexity of $O(N^3)$.

14.7.2 Illustrative examples

To illustrate some results for this covariance estimation, the transformations computed from four sets of feature correspondences are displayed in Figure 14.7, along with their 2-d uncertainty ellipses for $[x_i^\star \; y_i^\star]^\top$ and the densities of $\phi_i^\star$. These examples illustrate some interesting properties of the resulting uncertainty.

Firstly, the uncertainty in the orientation $\phi_i^\star$ strongly depends on the spatial distribution of the features, since more precise estimations can be made from features distributed over larger areas. This can be clearly observed by comparing the two cases shown in Figure 14.7(b)–(c). Secondly, the uncertainty in the 2-d coordinates of $\mathbf{q}_i^\star$ decreases with the number of features N only for very low values of N. This can be explained by the first term in C_{11} and C_{22}, that is, $\frac{2}{N}$, becoming negligible, whereas the second term does not decrease for incresingly larger values of N.

14.7.3 Validation

In order to validate our model of the covariance $\mathbf{Q}_i$, we have evaluated the Kullback-Leibler divergence between our model and the covariance obtained from a Monte-Carlo simulation comprising 6 pairs of correspondences between randomly located features corrupted with Gaussian noise. The results, in Figure 14.8, validate our derivation since the experimentally obtained covariance approaches the theoretical model as the number of Monte-Carlo trials increases.

14.8 Experimental evaluation and discussion

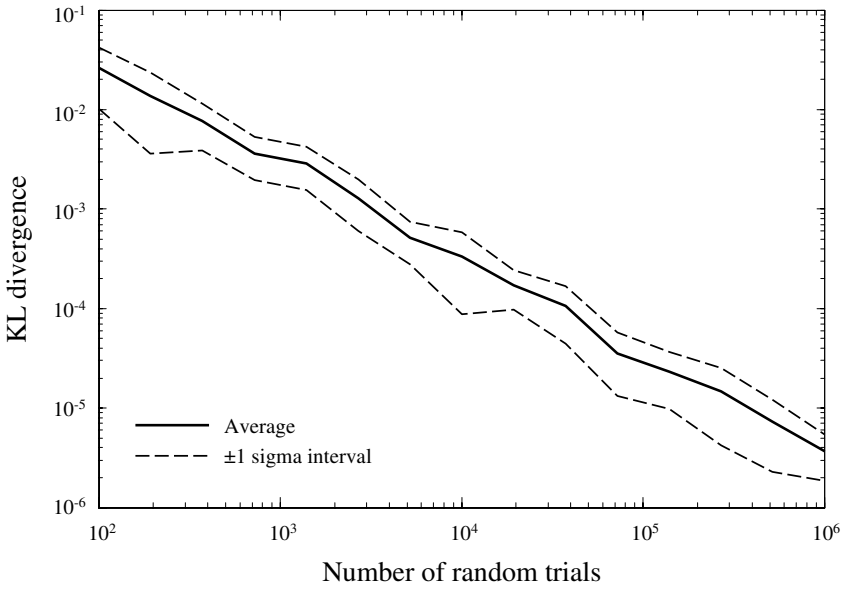


Figure 14.8: The Kullback-Leibler divergence between our theoretical model for the covariance $\mathbf{Q}_i$ and its value from a Monte-Carlo simulation for an increasing number of trials. Confidence intervals are shown for the KLD since the values at each point were computed for 50 different maps generated by randomly positioned features. These results represent an excellent agreement between the real covariance and the derived theoretical model.

In this section we present experiments aimed at testing the robustness of our approach against errors and noise, and also its dependability in the context of loop closure detection. For all these results we have employed the Harris corner detector and the linear-polar descriptor to establish correspondences between grid maps with a resolution of 10cm.

14.8.1 Performance under errors and noise

Maps built by a mobile robot at different moments may present significant differences due to both dynamic objects and errors in the robot localization while mapping. To quantify the accuracy of our method against such differences we have matched a reference map, built from real data, to a transformed one with known ground-truth translation and rotation – see the left column in Figure 14.9.

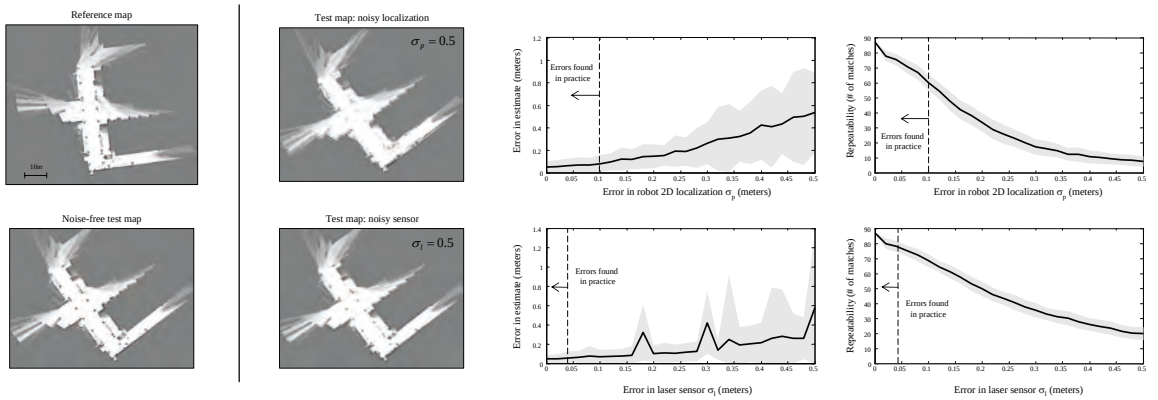


Figure 14.9: Characterization of our method under the presence of localization errors (σ_p) and laser sensor errors (σ_l). The average error in the map transformation and the average number of matches between the pair of maps are shown by the thick plot, while the $\pm$1-sigma confidence intervals are represented by the shaded region.

We have evaluated two sources of errors. Firstly, the estimated robot path in the environment (which in turn determines the accuracy of the map itself [Gut99]) has been deliberately corrupted by Gaussian noise with a standard deviation of σ_p. As σ_p increases, so does the degradation of the test map, as illustrated in the top row of the figure. It can be seen how the corresponding errors in the map transformation as detected by our method increases with larger σ_p, which is explained by both the more erroneous locations of detected features and their more reduced repeatability, shown in the rightmost column of the figure. Note that repeatability is a desired property of any feature detector since it assures that the same physical point is detected in two different maps in spite of potential changes in orientation or minor differences in the feature surroundings.

Secondly, we also evaluated the effects of noise in the laser scanner ranges, characterized by a standard deviation of σ_l. As shown in Figure 14.9, our method is less sensitive to this kind of error (in comparison to the localization error σ_p). One likely reason for this behavior is that the preprocessing of map images smooth out part of the noisy measurements.

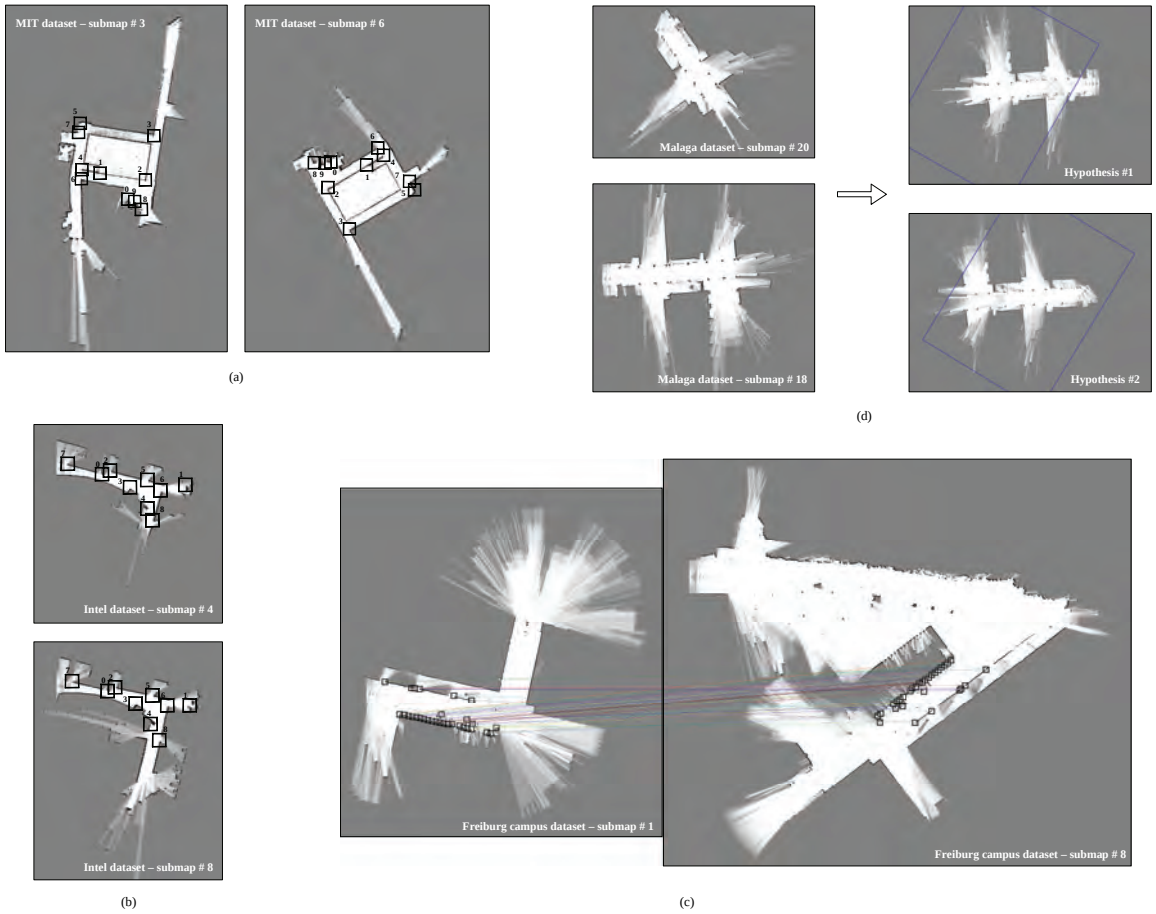


Figure 14.10: (a)–(c) Some examples of map-to-map matchings as detected by the proposed method. (d) A pair of submaps for which a multi-modal transformation is detected. The two different hypotheses are represented by the overlay of the submap #20 over submap #18 in the right hand images.

We must remark that the error and noise levels probed in this characterization are much higher than those expected in real-world conditions (the ranges of realistic values are marked in the graphs). Therefore, the errors of our method under normal conditions can be expected to be below 10cm, approximately.

14.8.2 Performance in loop-closure detection

The following benchmark characterizes the performance of our method in its natural application to hierarchical SLAM [Bla07b, Est05], that is, in detecting loop closures from local, metric submaps. For this aim we have selected four publicly available datasets. Three of them, the Freiburg campus dataset, the Intel dataset and the MIT dataset are published in the Radish repository [How03], while an additional dataset was collected by the author at the Málaga campus (see Figure 14.10 for example submaps from each dataset).

All these datasets have been processed within our HMT-SLAM framework described in Chapter 12. In this framework, the original sequence of robot observations is grouped into segments of consecutive observations (the submaps) according to a natural metric of similarity (refer to Chapter 13). For convenience, we disabled topological loop closure detection in this framework to obtain the raw sequence of submaps in each dataset. Therefore, among them some areas will appear several times corresponding to loop closures that should be detected by our method.

The so obtained set of 59 submaps is an ideal testbed for the method proposed in this chapter, since we can now try to match each submap with the rest, including those in different datasets (from which no valid transformation should result). The detailed results of executing the 1711 map-to-map matchings are shown in Figure 14.11, where each entry in the table specifies the outcome from our method and whether the two

maps actually do correspond or not, that is, it shows the loop closure ground truth (obtained by human inspection). Notice that there are two possible kinds of errors in this experiment: false positives (our method detecting a loop closure that does not really exist) and false negatives (where a real loop closure is overlooked). Due to the nature of SLAM, the latter is far more important, since missing a loop closure may degrade the global map, while a false positive may be easily rejected by metric information in the hierarchical map.

As can be seen in the figure, our method correctly detects as non-matchings virtually all the cases where each submap belongs to a different dataset. Two datasets deserve additional attention. Firstly, the Intel dataset leads to several false positives, which is explained by the symmetry of the environment, i.e. all its submaps are very similar. Secondly, the Málaga dataset also suffers from many false positives, most of them attributable to the environment, consisting of an array of three exactly identical buildings.

The overall performance is also summarized in Table 14.1, where for the sake of a fair validation we do not count the elements in the main diagonal of Figure 14.11 (matching each submap to itself), which were correctly detected by our method. It is remarkable that only one loop closure out of 41 was not recognized (a $\sim 2.4\%$ fail rate). We also show in the table the ratio of false positives modified by disregarding the errors clearly attributable to a real repetitive environment, not to errors in our detection method.

Regarding the computation time of this benchmark, it took 1740 seconds to compute the 1711 matchings in a Pentium Core Duo @ 2.2GHz (using a single execution thread), yielding an average 1.02 seconds per match.

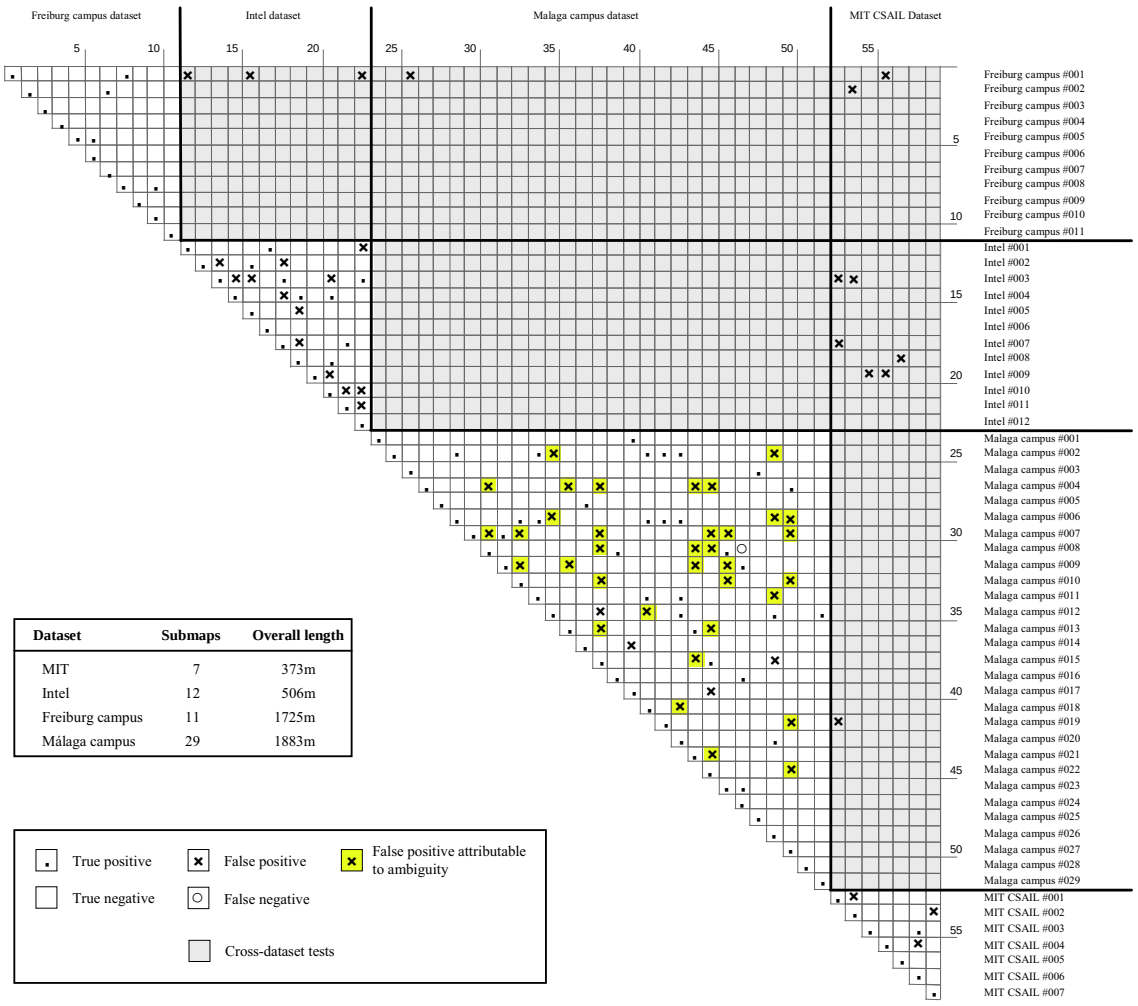


Figure 14.11: Results of the loop closure benchmark. The submaps corresponding to each of the two datasets have been separated by thick lines and inter-dataset blocks have been shaded for clarity.

Table 14.1: Results for the loop-closure detection benchmark.

	Result	Disregarding ambiguity
True positives	97.56% (40/41)	·
False positives	3.47% (58/1670)	1.38% (23/1670)
True negatives	96.53% (1612/1670)	98.62% (1647/1670)
False negatives	2.44% (1/41)	·

14.8.3 Discussion

In this chapter we have proposed a new approach to grid matching, based on existing computer vision techniques (detectors and descriptors) and providing the modifications required by the ambiguity typically found in our problem by means of a multi-hypothesis RANSAC stage. The resulting method has been demonstrated to assess a 97% success ratio in detecting loop closures while also being reliable against sensor noise and errors in the robot positioning.

Part IV

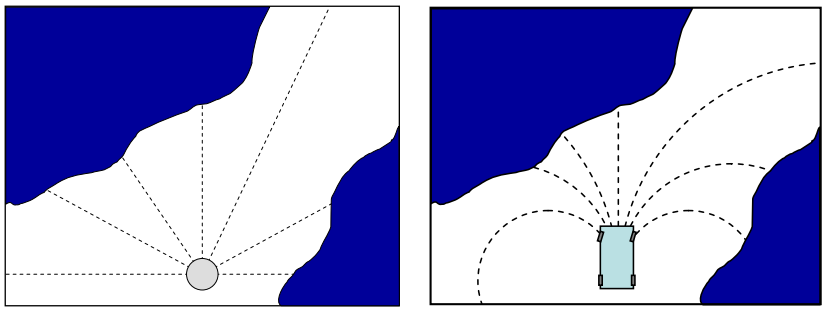
Mobile robot navigation

CHAPTER 15

OVERVIEW

15.1 Obstacle representation in robot navigation

Along with localization and mapping, autonomous and safe navigation is undoubtedly one of the issues which needs to be rigorously solved for mobile robots to leave research labs and become more practical. For achieving this purpose, it is essential to account for some spatial representation of obstacles where collision-free paths could be found efficiently. This problem has been extensively studied by the robotics community and has traditionally led to two different research areas. On the one hand we have *motion planning* approaches, where an optimal path is computed for a known scenario and a target location. The Configuration Space (C-Space) [LP83] has been successfully employed as representation in this scope: in C-Space the robot can be represented as a single point in the high-dimensionality space of its degrees of freedom. On the other hand, some navigation approaches deal with unknown or dynamic



(a) (b)

Figure 15.1: (a) Holonomic robots can move following straight lines without restrictions, while (b) realistic non-holonomic robot can only move following sequences of circular arcs.

scenarios, where motion commands must be periodically computed in real-time during navigation (that is, there is no planning). Under these approaches, called *reactive* or *obstacle avoidance* methods, the navigator procedure can be conveniently seen as a direct mapping between sensor readings and motor actions [Ark98]. Although reactive methods are quite efficient and have simple implementations, many of them do not work properly in practical applications since they often rely on too restrictive assumptions, like a point or circular representation of robots or considering movements in any direction, that is, ignoring kinematic restrictions. C-Space is not an appropriate space representation for reactive methods due to its complexity, which prohibits real-time execution. Hence simplifications of C-Space have been proposed specifically for reactive methods. Finally, combinations of the two above approaches have also been proposed [Kha97, Lam04, Qui93], which usually start computing an optimal planned path based on a known static map, and deform it dynamically to avoid collision with unexpected obstacles. These hybrid approaches successfully solve the navigation problem in many situations, but purely reactive methods are still required for partially known or highly dynamic scenarios, where an a priori planned path may need excessive deformation to be successfully constructed by a hybrid method.

In this thesis, the stress is on pure reactive methods, which aim at reactively driving a kinematically-constrained, any-shape mobile robot in a planar scenario. This problem requires finding movements that approach the target location while avoiding obstacles and fulfilling the robot kinematic restrictions. In particular, the contribution presented in Chapter 16 is related to the process for detecting free-space around the robot, which is the basis for a reactive navigator to decide the best instantaneous motor action. For this task, existing methods consider certain families of simple paths for measuring obstacle distances (i.e. they sample the free-space). These families of paths, or *path models*, must be considered not as planned paths but as artifacts for taking nearby obstacles into account. All existing reactive methods use path models that are an extension of the robot short-term action, as illustrated in Figure 15.1: for holonomic robots that can freely move in any direction, straight lines are used, while for non-holonomic robots virtually all existing methods employ circular arcs.

Straight and circular paths, used in previous reactive methods, are just two of the the infinity of path models that could be followed by a robot in a memoryless system, that is, reactively. It is clear that considering other path models can be more appropriate to sample the free-space than using the classic straight or circular models only. We shed light into this issue through the example in Figure 15.2, where a robot (reactively) looks for possible movements. If a single circular path model is used for sampling obstacles as in Figure 15.2(a), it is very likely that the obstacle avoidance method overlooks many good potential movements – notice that any reactive method must decide according solely to the information that path models provide about obstacles. In contrast, using a diversity of path models, as the example shown in Figure 15.2(b), makes much easier to find better collision-free movements.

A fundamental point in the process of using path models to sample obstacles is that *not any arbitrary* path model is suitable for reactive navigation, since it must assure

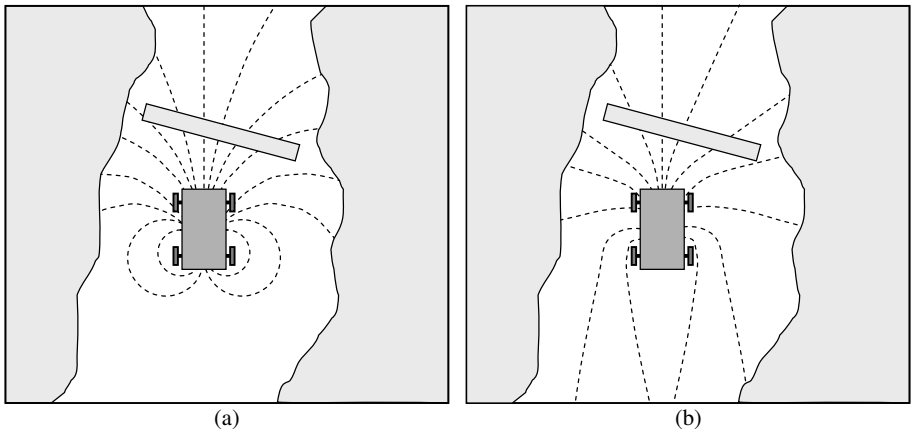


Figure 15.2: Reactive methods can take obstacles into account through a family of paths, typically circular arcs (a). However, we claim that other possibilities may be useful for finding good collision-free movements, as the path family shown in (b).

that the robot kinematic constrains are fulfilled while still being able of following the paths in a memory-less fashion. Therefore, it is worth discussing the properties of trajectories that fulfil this condition (see §16.3.1), an important reflection that cannot be found in the literature.

15.2 Survey of related works

Next, we examine the space representations typically employed in mobile robot motion planning and collision avoidance, and put them in contrast with the approach explained in detail in the next chapter.

C-Space has been extensively used in many fields, including robotic manipulators [LP87], maneuver planning [Lat91], and mobile robot motion planning [Mur00]. In spite of recent advances towards speeding up its computation [Zha06], the complexity derived from its high dimensionality makes C-Space not applicable to real-time reactive navigation. The problem can be visualized with the example of Figure 15.3(a),

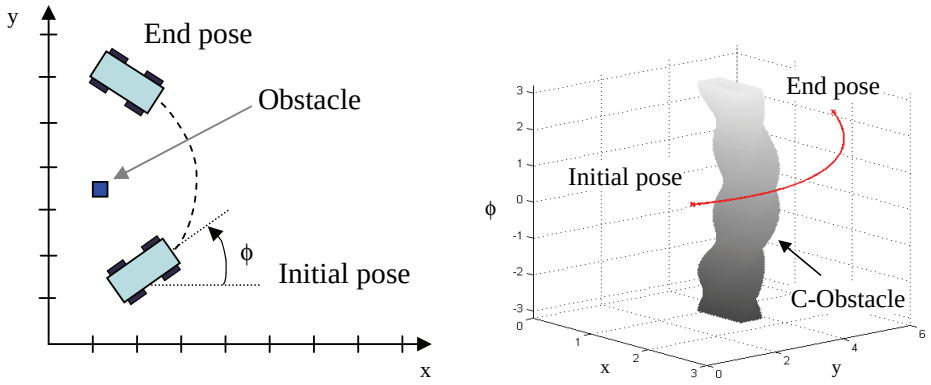


Figure 15.3: (a) A robot and a path avoiding an obstacle are represented here from a top view. (b) In C-Space the robot is now represented as a point, and the path becomes a 3D curve.

where it is shown a path for a robot in a planar scenario with an obstacle. This situation can be translated into C-Space as an obstacle that implicitly holds the robot shape (C-Obstacle). Then the navigation problem becomes that of finding a path in this space for a point robot, as shown in Figure 15.3(a). Unfortunately, building the whole C-Obstacles and finding paths in this high dimensional space remains being a computationally too expensive process.

A first simplification for dealing with C-Space more efficiently is to assume a circular robot. Thus, C-Obstacles are no longer dependent on the robot orientation and the C-Space reduces to a planar space, the Workspace (WS). This space is employed by the well known potential field methods (see [Cho05] for a review of these methods), like the VFF [Bor89], VFH [Bor91], and others [Had98, Bal93]. Other reported methodologies are based on neural nets [Pal95] and, more recently, the Nearness-Diagram (ND) approach [Min04], which relies on a divide-and-conquer strategy that defines a set of different states according to the arrangement of nearby obstacles. All these methods

deal with circular shaped robots, a too restrictive assumption for many real-life situations. For example, if a robotic wheelchair were assumed to be circular, it would never pass through a narrow doorway.

Most of the approaches that both deal with any-shape robots and take into account their kinematic restrictions propose working with another less limiting simplification of C-Space: the velocity space [Arr02b, Fei94, Ram01, Sch98, Sim96], or V-Space for short. For mobile robots of our interest, V-Space represents the space of the potential linear and angular robot velocities, hence the next movement can be chosen as a point in V-Space that results in constant curvature paths (i.e. circular paths). Notice that other methods that decide each robot action based on an evaluation of circular arcs, such as [Bon01], can be also classified as relying on V-Space since each circular arc maps into one line in this action space. A common feature in many V-Space methods is the inclusion of a dynamic window [Fox97], which restricts the range of reachable velocities to that compatible with the robot maximum acceleration. On the other hand, an important limitation is that, although many obstacles may be sensed, not all of them are exploited: only those ones falling into the robot dynamic window for the next step are considered for choosing the instantaneous motion command. It is clear that better paths could be found if more comprehensive obstacle information was taken into account, which is the central idea of the approach presented in the next chapter.

In addition to the utilization of a dynamic window, most V-Space approaches use only the family of circular paths to sample the free-space, which entails the risk of not detecting many free-space areas. There are some exceptions [Ram01, Xu02] that make use of straight paths, but this model is not appropriate for most actual mobile robots. Only these two path models have been reported in the reactive collision avoidance literature.

While a generic path can only be described in the three-dimensional C-Space (2-d position plus heading), a circular path can be defined through two parameters: the path curvature and the distance along the arc. Upon this parametrization, a *Trajectory Parameter space* (or TP-Space for short) was proposed in [Min06] as an elegant and mathematically sound alternative to V-Space: if the navigation is carried out in the 2-d TP-Space, the robot can be treated as a free-flying-point. That work demonstrates that navigation in a parameterized space allows us to decouple the problems of kinematic restrictions and obstacle avoidance. However, this approach has never been extended neither to cope with other path models apart from the circular one, nor to a number of different transformations, alternatives explored in this thesis.

To further clarify the relationship between the different existing representation spaces, please refer to Figure 15.4, where TP-Spaces appear as a generalization of spaces such as WS and V-Space. However, it must be remarked that C-Space is the most general space representation, but at the price of an elevated computational cost due to its high dimensionality.

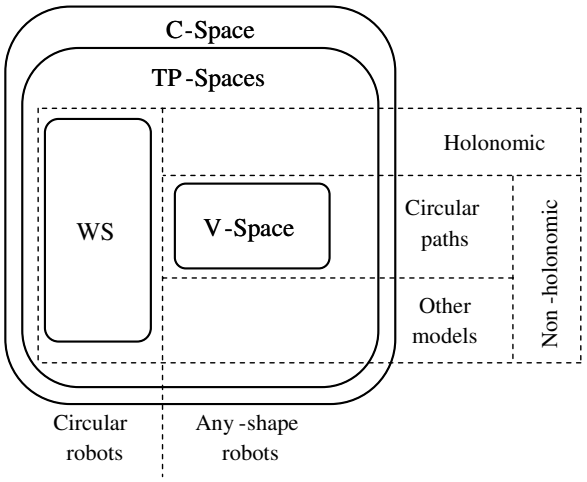


Figure 15.4: A summary with the classification of space representations used in obstacle avoidance methods.

CHAPTER 16

REACTIVE NAVIGATION BASED ON PTGS

16.1 Introduction

This chapter addresses the problem of reactive navigation for any-shape, kinematically-constrained mobile robots in planar scenarios. This requires efficiently finding movements that approach the target location while avoiding obstacles and fulfilling the robot kinematic restrictions.

Our main contribution here is about the process for detecting free-space around the robot. As explained in Chapter 15.1, existing reactive methods consider straight and circular path models for measuring obstacle distances (see Figure 16.1), although different models have been studied in the field of motion planning, obtaining interesting results regarding shortest paths [Lau93, Sou96, Ven99]. Our approach allows some of

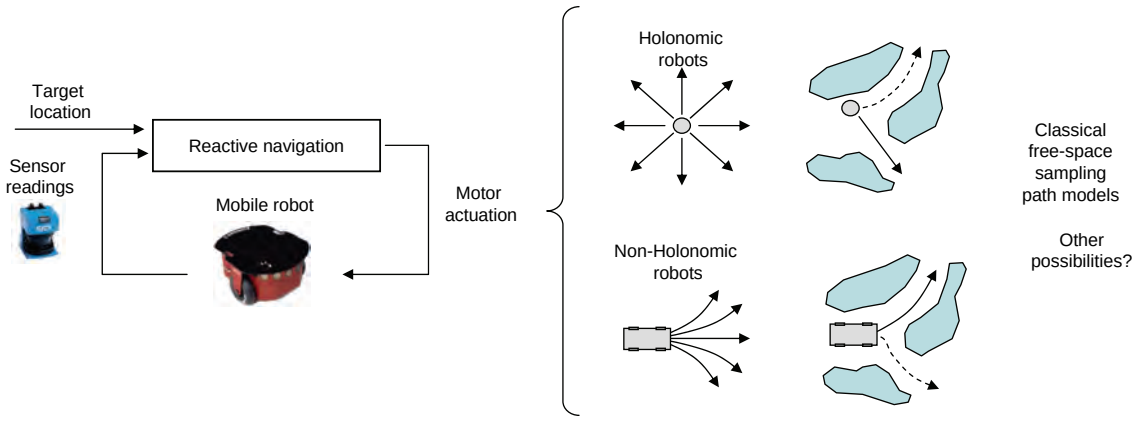


Figure 16.1: Reactive navigation methods periodically map sensor readings to motor actuations in order to avoid obstacles and to reach a target location. For taking obstacles into account, straight and circular sampling path models have been used for holonomic and non-holonomic robots, respectively. However, other valid possibilities exist (dashed curves are examples) which provide the robot with a more comprehensive free-space detection.

those results to be integrated into a reactive navigation system for the first time.

As an example of the decisions made during reactive navigation, consider the robot in Figure 16.2(a), which must decide its next movement from a family of circular arcs, each one giving a prediction for the distance-to-obstacles. Since reactive navigation is a discrete time process, the decision will be taken iteratively in a timely fashion, though at each time step the family of paths will be considered starting at the *current pose* of the robot. The central issue here is that, implicitly, it is assumed that if the robot chooses one path at some instant of time, at next time step it will have the possibility of *continuing along the same trajectory*. Otherwise, the prediction of distance-to-obstacles would be useless since foreseen trajectories will never be actually followed. In the case of circular arcs, this property indeed holds, as illustrated in the example in Figure 16.2(b). The main contribution of the present work is a detailed formalization of this and other properties that need to hold for a path model being applicable to obstacle avoidance.

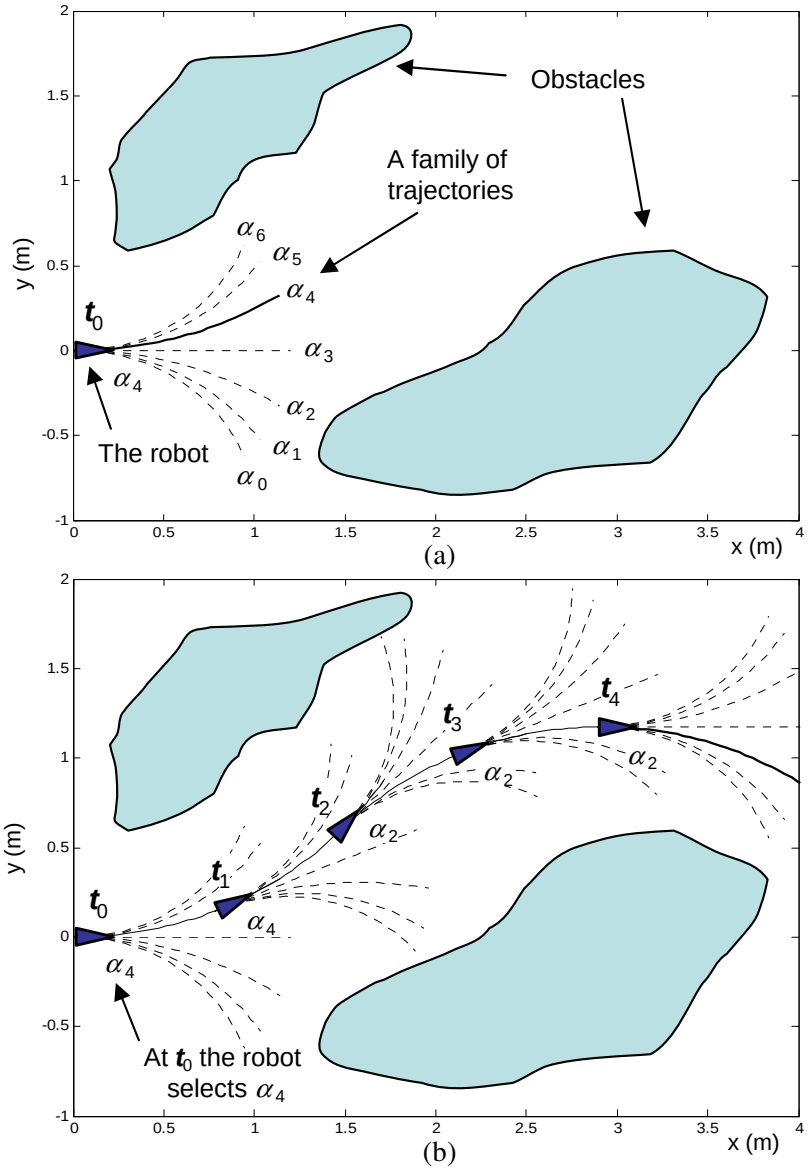


Figure 16.2: A schematic representation of the process involved in reactive navigation. At each time step, the robot employs a family of paths to sample the obstacles in the environment, then chooses the most convenient action according to that information. It must be highlighted the important implicit assumption in the process: that the robot will be able to continue trajectories chosen at previous time steps. Since this does not hold in general for all path models, it is presented in the text a template for path models that are proven to fulfill this requirement.

As reported previously elsewhere [Bla06c, Bla08f], the problems of kinematic restrictions and obstacle avoidance can be decoupled by using path models to transform kinematic-compliant paths and real-world obstacles into a lower complexity space, i.e. a Trajectory Parameter Space (TP-Space). The transformation is defined in such a way that the robot can be considered as a free-flying-point in the TP-Space since its shape and kinematic restrictions are already embedded into the transformation process. We can then entrust the obstacle avoidance task to any standard holonomic method operating in the transformed space. This idea was firstly introduced by Minguez and Mintano in [Min02] with the study of the Ego-Kinematic Transformation (EKT), and has subsequently evolved in a series of works [Min06, Min09].

The main contribution of the work discussed here [Bla08f] is the extension of the EKT approach [Min06] with the generalization of path models through a novel tool called Parameterized Trajectory Generator (PTG), which permits us to handle any number of transformations instead of just the one corresponding to circular arcs.

It is also proposed a navigation system to manage simultaneously multiple paths, each corresponding to a different PTG. This system faces the dynamic selection of the best alternative at each instant of time (for instance, in terms of path length and clearance), which yields a more powerful approach than traditional methods relying on circular paths only.

To sum up, the proposed solution to reactive navigation presents the following features:

1. The problems of obstacle avoidance and kinematic restrictions are decoupled like in [Min06], but now in a generalized way that allows a more effective detection of free-space.

2. Well-known, efficient reactive methods previously constrained to holonomic point

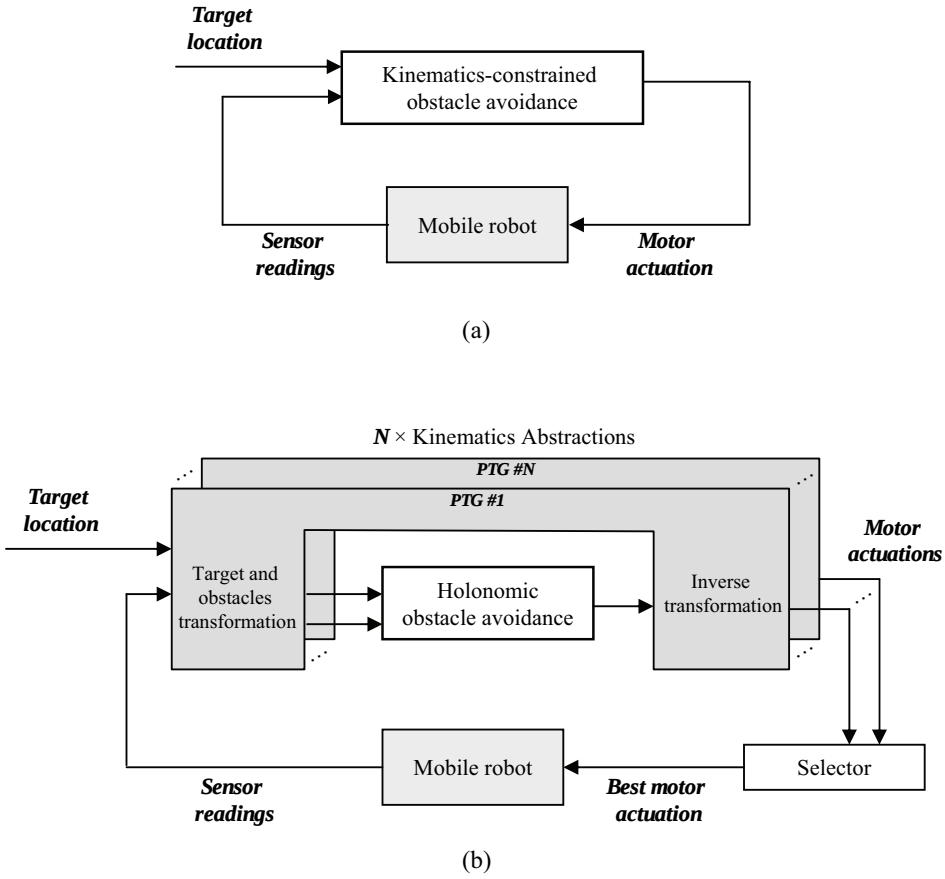


Figure 16.3: Instead of considering the robot kinematics into the reactive method itself like most existing approaches do (outlined in (a)), we propose here to use a number of different kinematics abstractions layers, as shown in (b). A simple holonomic method is executed for each one of the transformed scenarios.

(or circular) robots can be applied to any-shape, non-holonomic ones.

3. Using path models other than circular ones enables the robot to find better collision-free movements. Furthermore, a number of path models can be simultaneously used and the one that best suits to each specific situation can be dynamically chosen.

Following a representation similar to that in [Min09], Figure 16.3 represents both a classical reactive navigation method and our approach. As stated above, previous methods that consider the kinematic restrictions of mobile robots deal with them and

obstacle avoidance as a whole, which is illustrated in Figure 16.3(a). We propose to abstract the kinematic restrictions from the collision avoidance through a number of different transformations (PTGs). This idea is shown in Figure 16.3(b), where it can also be seen the selection step needed to choose the best transformation at each instant of time.

Regarding the outline of the rest of this chapter, we will first present the definition of parameters spaces, then employ it in §16.3 to formally define what a PTG is and its properties. Next, some examples of practical PTGs are presented in §16.4, and we finish with an in-depth discussion of practical issues related to the implementation of a PTG-based reactive navigation system and several experiments in both synthetic and real environments.

16.2 Trajectory parameter spaces (TP-Spaces)

In this section we first discuss the problem of how to measure distance to obstacles for a non-holonomic, any-shape robot, which is the basis for the definition of a TP-Space.

16.2.1 Distance-to-obstacles

Calculating distance-to-obstacles is an essential step in any reactive navigation algorithm since it provides the robot with information for choosing the next movement. To the best of our knowledge, all previous (reactive) works make an implicit assumption that has never been questioned: distance-to-obstacles (i.e. collision distances) are computed by means of a single fixed path model: either straight or circular, commonly depending on the robot being holonomic or not. Distances are then taken along those

2-d paths, though robot paths are actually defined as continuous sequences of locations and orientations, that is, as three-dimensional [1] curves in C-Space as seen in Figure 16.4(a). Thus, to be rigorous distance-to-obstacles should be directly measured in C-Space, through the mechanism described next.

By representing all the paths from a given path model simultaneously in C-Space it is obtained a 3-d surface, as the example in Figure 16.4(b). These surfaces will be referred to as *sampling surfaces*, since distance-to-obstacle can be computed by measuring the distance from the origin to the intersection of those surfaces with C-Obstacles. Next we can "straighten out" the surface into a lower dimensionality space where obstacle avoidance becomes easier, that is, a TP-Space. In this process the topology of the surface is not modified. Since we are proposing a diversity of path models to be used simultaneously, we will have different associated sampling surfaces in C-Space to compute distance-to-obstacles. The whole process is illustrated in Figure 16.4(c). Notice that the measurement of distances in C-Space combines linear and angular values, as is highlighted in Figure 16.4(a). This problem can be worked out by defining alternative non-Euclidean metrics as discussed later on.

16.2.2 Definition of TP-Space

We define a TP-Space as any two-dimensional space where each point corresponds to a robot pose in a sampling surface. It is convenient to consider points in a TP-Space by their polar coordinates: an angular component α and a distance d. In this way the angular coordinate has a closed range of possible values. The mapping between a TP-Space and a sampling surface is carried out by selecting an individual trajectory out from the family using the α coordinate, while d establishes the distance of the pose along

[1] We refer to C-Space as a 3-d space due to the three dimensions of its topological structure $R^2 \times S^1$, although this differs from the usual meaning of 3-d Cartesian spaces with R^3 structure.

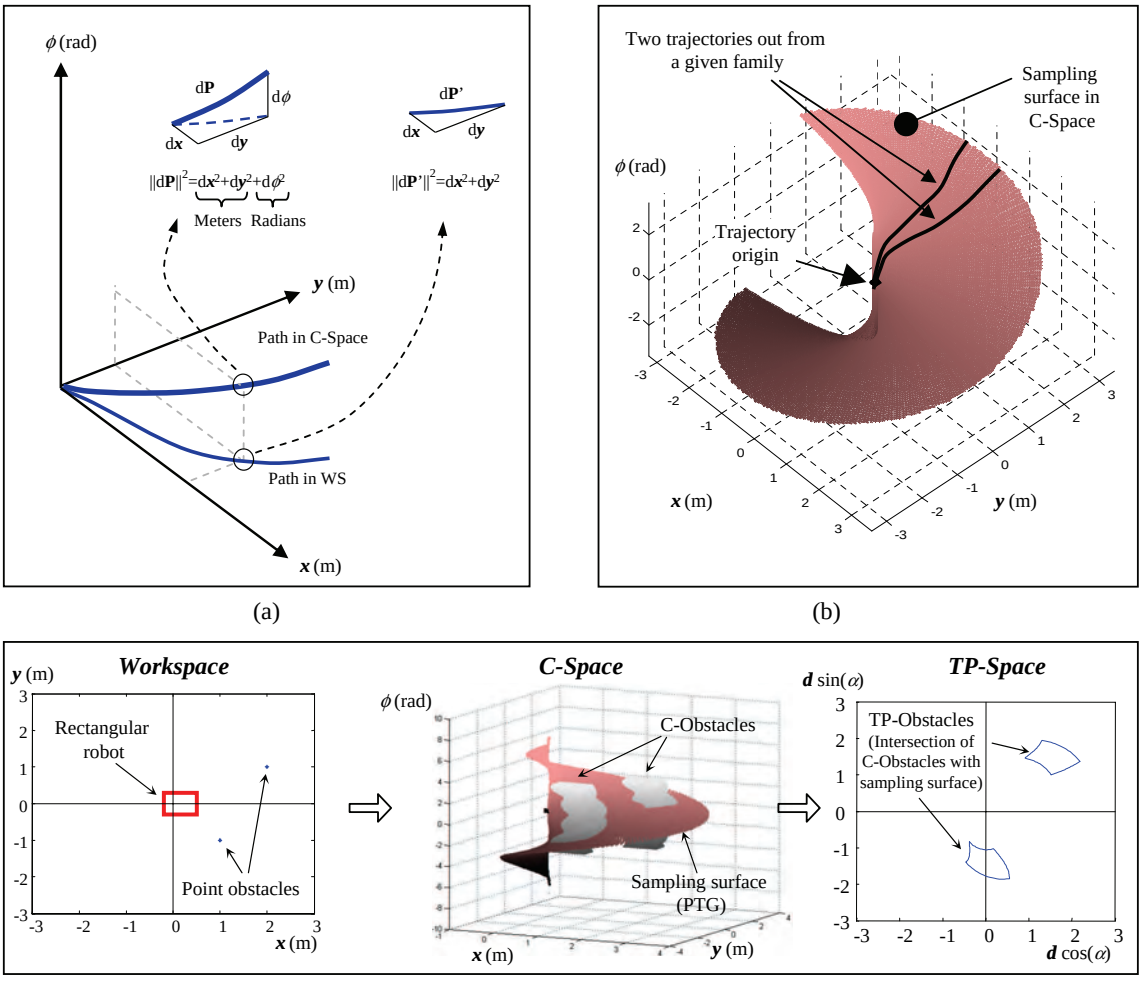


Figure 16.4: Our vision of how to measure distance to obstacles. (a) Robot paths can be visualized as curves in C-Space. (b) A family of parameterized paths (PTG) generates a 3-d surface that can be used to sample obstacles. (c) The transformation from real obstacles into TP-Space can be seen as the projection into TP-Space of the intersection of the sampling surface with C-Obstacles.

that selected trajectory. In the original work by Minguez and Montano [Min06], such a distance is measured as the Euclidean distance disregarding the robot orientation. Alternatively, it is proposed here to measure distances in a TP-Space through a non-Euclidean metric, directly along C-Space sampling surfaces.

The region of interest in TP-Space is a circle centered at the origin and of radius R_m (a constant that settles the collision avoidance maximum foresee range). We will refer to the TP-Space domain as the 2-d space $\mathbb{S} \times D$, with $\mathbb{S} =]-\pi, \pi]$ and $D = [0, R_m]$. Note as well that the transformation is applied at each iteration of the navigation process, thus for all our derivations the robot is always at the origin.

16.3 Parameterized trajectory generators (PTGs)

A PTG is any of the possible transformations that convert TP-Space points into robot poses, as depicted in Figure 16.5. In the following a more formal definition of a PTG is given and some of its properties are discussed.

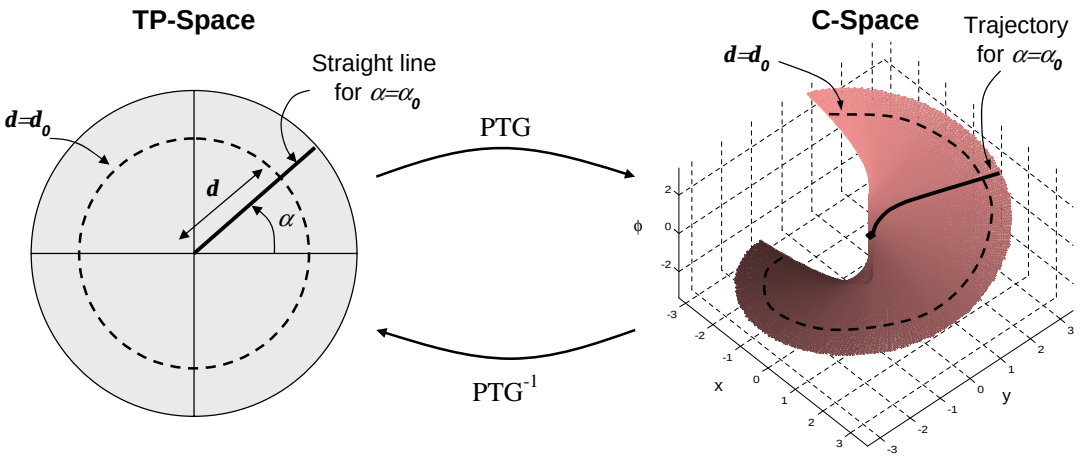


Figure 16.5: A point given by its polar coordinates (α, d) in TP-Space (above) represents a robot pose (x, y, ϕ) on a particular sampling surface in C-Space (below). The transformation is defined by a given PTG. The parameter α selects a particular path within a family, while d indicates the distance from the origin.

16.3.1 Basic definitions

We define a 2-d robot trajectory for a given parameter value α as:

$$\mathbf{q}(\alpha, t) = \begin{bmatrix} x(\alpha, t) \\ y(\alpha, t) \\ \phi(\alpha, t) \end{bmatrix} \quad , t \geq 0 \tag{16.3.1}$$

For realistic robots subject to non-holonomic constrains, trajectories are defined as the integration of their time derivative $\dot{\mathbf{q}}(\alpha, t)$, that is,

$$\mathbf{q}(\alpha, t) = \int_0^t \dot{\mathbf{q}}(\alpha, \tau) d\tau. \tag{16.3.2}$$

where it applies the initial condition $\mathbf{q}(\alpha, 0) = \mathbf{0}$ for any α.

Note that TP-Space is defined in terms of distance d (see §16.2.2) rather than time t, in which the kinematic equations are naturally defined. The reason for this change of variable is that we are interested in the geometry of paths, which remains unmodified if the velocity vector[2] $\mathbf{u}(\cdot)$ is multiplied by any positive scalar, an operation equivalent to modifying the speed of the robot dynamically. For example, it is common in navigation algorithms to adapt the robot velocities to the clearness of its surroundings.

Therefore, we define a PTG as:

$$PTG(\alpha, d) \doteq \mathbf{q}(\alpha, \mu_\alpha^{-1}(d)) \tag{16.3.3}$$

where the function $\mu_\alpha^{-1}(d)$ maps distances d to times t. It is proven below that this function exist and is well defined for many PTGs. Thus, a PTG is a mapping of TP-Space points to a subset of C-Space:

[2] Notice that we will use $\mathbf{u}$ to refer to the vector of robot (linear and angular) velocities, whereas the term $\dot{\mathbf{q}}$ (the time-derivative of the pose) gives us the velocities in a Cartesian coordinate frame, which are not of our interest here.

$$\begin{array}{rccc} PTG: & A \times D & \mapsto & \mathbb{R}^2 \times \mathbb{S} \\ & (\alpha, d) & \mapsto & \mathbf{q} \end{array} \tag{16.3.4}$$

In the common cases of car-like or differentially-driven robots the derivatives in Eq. (16.3.2) are given by only one set of kinematic equations:

$$\begin{aligned} \dot{\mathbf{q}}(\alpha, t) &= \mathbf{J}(\mathbf{q}(t, \alpha))\mathbf{u}(\alpha, t) \\ &= \begin{bmatrix} \cos(\phi(\alpha, t)) & 0 \\ \sin(\phi(\alpha, t)) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v(\alpha, t) \\ \omega(\alpha, t) \end{bmatrix} \end{aligned} \tag{16.3.5}$$

Here $\mathbf{u}$ is the vector comprising the linear (v) and angular (ω) velocities of the robot at each instant of time t and for each value of the PTG parameter α. The freedom for designing different PTGs is therefore bound up with the availability of different implementations of $\mathbf{u}(\cdot)$. In §16.4 some of the possible designs are discussed.

In spite of the fact *any* function $\mathbf{u}(\cdot)$ represents a kinematically valid path for a robot, which follows from Eq. (16.3.5) by definition, the present work is built upon the realization of not any arbitrary function leads to *valid* space transformations for obstacle avoidance methods. We specify next when such a transformation is valid for our purposes.

Definition. A space transformation between C-Space and TP-Space is said to be *valid* when it fulfills the following conditions:

- **C1.** It generates *consistent reactive trajectories*. All path models are not applicable to reactive navigation because of the memoryless nature of the movement decision process, as discussed in section 16.1.

- **C2.** It is *WS-bijective*. For each WS location (x, y), at most one trajectory can exist taking the robot to it, regardless of the orientation. Otherwise, the target position would be seen at two different directions (straight lines) in TP-Space – recall that a PTG maps straight lines of the TP-Space into trajectories of the C-Space.

- **C3.** It is *continuous*. Together with the last restriction, this condition assures that transformations do not modify the topology of the real workspace around the robot.

These three conditions hold for the case of classical circular arcs. An important theoretical contribution is the following theorem that proves that a broader variety of valid PTGs are indeed suitable to reactive navigation.

Theorem 16.3.1. *A sufficient, but not necessary condition for a PTG to be* valid *is its velocity vector* $\mathbf{u}$ *being of the form:*

$$\mathbf{u}(\alpha, t) = \begin{bmatrix} v_m \cdot f_v\left(a\alpha + b\phi(\alpha, t)\right) \\ \omega_m \cdot \left(a\alpha + b\phi(\alpha, t)\right) \end{bmatrix} \tag{16.3.6}$$

where v_m *and* ω_m *settle the desired maximum linear and angular velocities in absolute value, respectively,* $f_v(\alpha, t)$ *is any Lipschitz continuous function which evaluates to non-zero over the whole domain, and* a, b *are arbitrary constants with the restrictions* $0 < |a/b| \leq 1$ *and* $b < 0$. *Furthermore, a velocity vector of this form becomes fully defined by just specifying its value for* $t = 0$.

The following section is devoted to a detailed analysis of PTGs in this form and to prove our claim of they always being valid in the sense that they fulfill all the conditions listed above.

16.3.2 Proofs

We start by defining the function $\mu_\alpha(t)$ as the distance traveled by the robot along trajectories in C-Space from the origin and until the instant t, that is:

$$\mu_\alpha(t) = \int_0^t \left\| \frac{\partial \mathbf{q}(\alpha, \tau)}{\partial \tau} \right\| d\tau \tag{16.3.7}$$

where the norm could be the Euclidean distance, though we will employ here a custom metric introduced in [Bla08f] which accounts for robot turns more properly through a constant ρ that roughly represents the robot radius, leading to:

$$\mu_\alpha(t) = \int_0^t \left(v(\alpha, \tau)^2 + \rho^2 \omega(\alpha, \tau)^2 \right)^{\frac{1}{2}} d\tau \tag{16.3.8}$$

Then, we can state the following lemma about the existence of $\mu_\alpha^{-1}(d)$, required in Eq. (16.3.3) for the definition of PTGs.

Lemma 16.3.2. *The function $\mu_\alpha : t \mapsto d$ is continuous and its inverse $\mu_\alpha^{-1} : d \mapsto t$ is well-defined for any $t \geq 0$.*

Proof. The first part, proving the continuity of $\mu_\alpha(t)$ is trivial since the function is defined as an integral, thus it always has a derivative. Next, it can be seen that the function is strictly increasing due to its derivative being the norm of $\dot{\mathbf{q}}$, which in general is non-negative, but given the hypothesis from theorem 1 of f_v evaluating to non-zero over all the domain, the case $\dot{\mathbf{q}} = 0$ can be ruled out. Being continuous and strictly-increasing, $\mu_\alpha(t)$ becomes bijective for any $t \geq 0$, thus its inverse is well-defined. □

An important feature of any valid PTG is that different values of α must generate unique trajectories (see condition C2), which is assured by the following lemma.

Lemma 16.3.3. *Provided $b < 0$ and $0 < |a/b| \leq 1$, each value $\alpha \in \mathbb{S}$ determines a unique trajectory passing through the origin with its heading tending to $-\alpha \cdot a/b$ as $t \to \infty$.*

Proof. Since $\dot{\mathbf{q}}(\alpha, t)$ is Lipschitz continuous, and given the initial conditions $\mathbf{q}(\alpha, 0) = \mathbf{0}$ for any value of α, there exists only one trajectory for each α value [Eva92], which is determined by the value of $\dot{\mathbf{q}}(\alpha, 0) = [v(\alpha, 0)\ \omega(\alpha, 0)]^\top$. By hypothesis from theorem 1, we have $\omega(\alpha_1, 0) \neq \omega(\alpha_2, 0)$ for any $\alpha_1 \neq \alpha_2$ as long as $a \neq 0$ (refer to Eq. (16.3.6)), thus the uniqueness of each trajectory is assured.

Regarding the limit of the robot heading $\phi(\alpha, t)$, we can solve the differential equation of the kinematic model in Eq. (16.3.5) for this term, that is:

$$\begin{aligned} \dot{\phi}(\alpha, t) &= \omega(\alpha, t) \\ &= \omega_m \cdot (a\alpha + b\phi(\alpha, t)) \end{aligned} \tag{16.3.9}$$

which can be straightforwardly solved giving us:

$$\phi(\alpha, t) = -\alpha \frac{a}{b} \left(1 - e^{\omega_m b t}\right) \tag{16.3.10}$$

The parameter b determines the evolution of the heading over time. Since the robot heading must be bounded to the domain of $\mathbb{S}^1$, we discard the values of $b > 0$. The case $b = 0$ must be avoided as well since in that case Eq. (16.3.10) is not defined. Therefore, for the valid values $b < 0$, the heading converges to:

$$\lim_{t \to \infty} \phi(\alpha, t) = -\alpha \frac{a}{b} \tag{16.3.11}$$

Notice that the condition $0 < |a/b| \leq 1$ assures $\phi(\cdot)$ will always remain within its valid domain $\mathbb{S}$, which was the claim of this lemma. □

We address next the fundamental property of generated paths being consistent reactive trajectories – as stated by condition C1. The geometrical meaning of this property was discussed in section 15.1, and is now stated formally as follows.

Lemma 16.3.4. *For any $\alpha \in \mathbb{S}$ and $t_0 \geq 0$, there exists one $\alpha' \in \mathbb{S}$ such as:*

$$\mathbf{q}(\alpha, t_0 + t) = \mathbf{q}(\alpha, t_0) \oplus \mathbf{q}(\alpha', t) \quad , \quad \forall t \geq 0 \tag{16.3.12}$$

with α' being a function of α and t_0 (denoted, for instance, $A(\alpha, t_0)$) and where the $\oplus$ operator stands for 2-d pose composition (see appendix A).

Proof. It can be trivially shown that this statement holds for $t = 0$, when Eq. (16.3.12) becomes:

$$\mathbf{q}(\alpha, t_0) = \mathbf{q}(\alpha, t_0) \oplus \mathbf{q}(\alpha', 0) = \mathbf{q}(\alpha, t_0) \oplus \mathbf{0} = \mathbf{q}(\alpha, t_0) \tag{16.3.13}$$

Now, since both trajectories $\mathbf{q}(\alpha, t_0 + t)$ and $\mathbf{q}(\alpha, t_0) \oplus \mathbf{q}(\alpha', t)$ pass through a common point in C-Space at $t = 0$, it is enough to prove that their derivatives $\dot{\mathbf{q}}$ are identical at that instant for lemma 16.3.3 to imply that both trajectories coincide for any $t > 0$.

Taking into account the change of coordinates introduced by the pose composition operator, the condition of both time derivatives $\dot{\mathbf{q}}(\cdot)$ must coincide amounts to their velocity vectors $\mathbf{u}(\cdot)$ being identical at $t = 0$, that is, we must prove:

$$u(\alpha', 0) = u(\alpha, t_0) \tag{16.3.14}$$

By noticing from Eq. (16.3.6) that $\mathbf{u}$ is a function of the term $a\alpha + b\phi(\alpha, t)$, the above condition can be rewritten as:

$$\begin{aligned} a\alpha' + b\underbrace{\phi(\alpha', 0)}_{0} &= a\alpha + b\phi(\alpha, t_0) \\ aA(\alpha, t_0) &= a\alpha + b\phi(\alpha, t_0) \\ \rightarrow A(\alpha, t_0) &= \alpha + \frac{b}{a}\phi(\alpha, t_0) \end{aligned} \tag{16.3.15}$$

which gives us an expression for the $A(\alpha, t_0)$ function that fulfills this lemma statement.

□

It is interesting to highlight that the resulting expression for $A(\alpha, t)$ indicates that α' is well-behaved, in the sense that it never exceed the limits $]-\pi, \pi]$. It also reveals that all trajectories eventually become a straight path, as can be seen by taking the limit:

$$\lim_{t\to\infty} A(\alpha, t) = \alpha - \frac{b}{a}\alpha\frac{a}{b} = 0 \tag{16.3.16}$$

where the fact that $\alpha = 0$ generates a straight trajectory follows from the PTG design equations in theorem 1. Note how the final part of all the trajectories being identical to one of them aligns perfectly with our goal of *consistent reactive trajectories* (condition C1).

Finally, the last requisite of a valid PTG (condition C3) is to generate *continuous* sampling surfaces in C-Space, that is, the function $PTG(\alpha, d)$ must be continuous.

Lemma 16.3.5. *Given the hypotheses of theorem 1, $PTG(\alpha, d)$ is a 2-manifold of C-Space with boundaries, and is continuous and derivable over the whole domain $(\alpha, d) \in \mathbb{S} \times D$.*

Proof. Firstly, due to lemma 16.3.2 it is enough to prove the continuity and differentiability of $\mathbf{q}(\alpha, t)$ since the mapping between distances d and times t conserves those properties of $\mathbf{q}(\cdot)$.

We show next that $\mathbf{q}(\alpha, t)$ has well-defined derivatives, which in turns implies it is continuous. For the case of $\frac{\partial \mathbf{q}(\alpha,t)}{\partial t}$ the proof is trivial since by definition this derivative is given by Eq. (16.3.5).

The derivation of $\frac{\partial \mathbf{q}(\alpha,t)}{\partial \alpha}$ is more involved. It is illustrative to keep Figure 16.6 as a reference through the following derivations to clarify the geometrical meaning of each term. Let dt be an infinitesimal increment in time, and (α, t) some fixed point in the domain of TP-Space. Then, using lemma 16.3.4 we can rewrite $\mathbf{q}(\alpha, t + dt)$ as:

$$\mathbf{q}(\alpha, t + dt) = \mathbf{q}(\alpha, dt) \oplus \mathbf{q}(\alpha', t) \tag{16.3.17}$$

where α' is given by:

$$\begin{aligned} A(\alpha, dt) &= \alpha + \frac{b}{a}\phi(\alpha, dt) \\ &= \alpha + \underbrace{\frac{b}{a}\omega(\alpha, 0)dt}_{d\alpha} \\ &= \alpha + d\alpha \end{aligned} \tag{16.3.18}$$

Making use of the definition of pose composition operators (see appendix A) we can rearrange Eq. (16.3.17) as follows:

$$\mathbf{q}(\alpha', t) = \mathbf{q}(\alpha, t + dt) \ominus \mathbf{q}(\alpha, dt) \tag{16.3.19}$$

The geometrical meaning of this operation is that, as illustrated in Figure 16.6(b), the curve $\mathbf{q}(\alpha', t)$ matches the curve $\mathbf{q}(\alpha, t)$ if translated and rotated to the pose

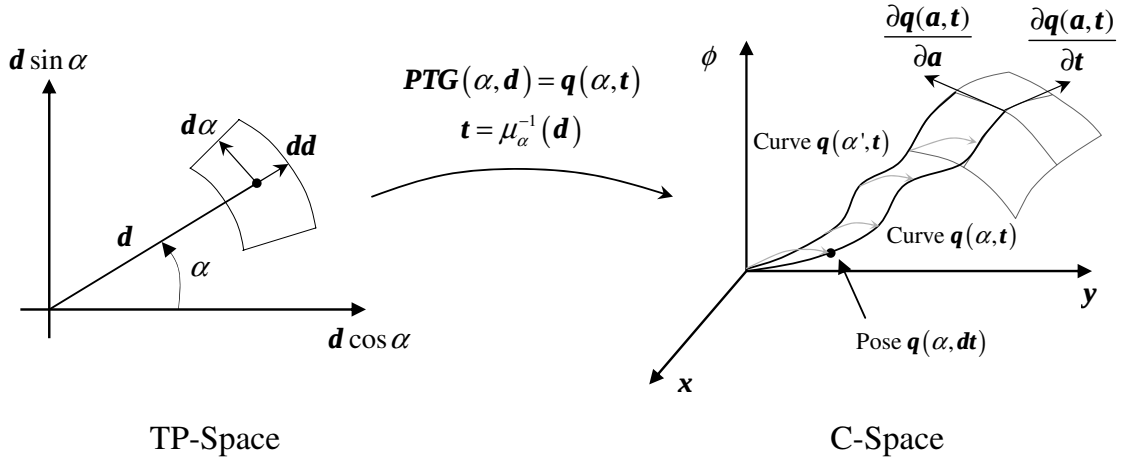


Figure 16.6: A schematic representation of the mapping a PTG performs between TP-Space points and C-Space robot poses. We represent the infinitesimal elements used in the proof of Lemma 16.3.5. Basically, the idea represented here is that the curve for the trajectory $\mathbf{q}(\alpha', t)$ matches precisely to the trajectory $\mathbf{q}(\alpha, t)$ if the coordinate origin of the former is changed to $\mathbf{q}(\alpha, dt)$ for some α' infinitesimally close to α.

$\mathbf{q}(\alpha, dt)$. As a result, this means that infinitesimal changes $d\alpha$ in a pair (α, t) leads to infinitesimal changes in $\mathbf{q}(\alpha, t)$ that can be written down as:

$$\mathbf{q}(\alpha + d\alpha, t) = \mathbf{q}(\alpha, t) \oplus \mathbf{J}(\phi(\alpha, t))\mathbf{u}(\alpha, t)dt \ominus \begin{bmatrix} v(\alpha, 0)dt \\ 0 \\ \omega(\alpha, 0)dt \end{bmatrix} \tag{16.3.20}$$

which follows from Eq. (16.3.19) and the definition of $\mathbf{q}$ as an integral of the velocity vector $\mathbf{u}$. Since the pose composition $\oplus$ and inverse composition $\ominus$ operators are both continuous and differentiable, it follows that the derivatives of $\mathbf{q}$ at the point (α, t) are well-defined, and thus it is continuous and differentiable as stated in the lemma.

Finally, given $\mathbf{q}(\alpha, t)$ is differentiable and so is $PTG(\alpha, d)$ at the whole domain of (α, d), the surface generated by a PTG can be seen as a 2-manifold with boundaries [Spa81]. □

16.4 Design of PTGs

16.4.1 Discussion

Not any arbitrary design function $\mathbf{u}(\alpha, t)$ leads to a valid PTG, since it must be somehow guaranteed that collision avoidance can be performed in the resulting transformed space by the reactive method running on it. Those methods have been designed to work in the WS, but they will be applied to TP-Space (which can be seen as a virtual WS). These requirements have been explored in detail in the previous section.

Besides these restrictions, other considerations may be also taken into account when designing a PTG, such as the robot speed limit or any other kinematic constraints. For example, a car-like robot will impose a minimum turning radius, which turns into a maximum angular-to-linear velocity ratio.

While circular paths (the classic path model) can be shown to fulfill the requirements, a proof for the general case can not be provided due to the lack of a generic analytical solution for Eq. (16.3.2). However, four different templates are presented next that guarantee that only valid PTGs are obtained from them. Three of them are combinations or simple arcs and the remaining one (named $\alpha - A$) is an instance of the generic template discussed in §16.3. These templates cover a sufficiently wide diversity of trajectories, as shown in a later section with several experiments.

The four templates are summarized in Figures 16.7–16.8. The rest of this section is devoted to discussing them. A common feature to all of the following design schemes is that a variety of PTGs can be generated from each one by choosing different values of the design parameters.

It must be remarked that these PTGs have potential applications not only to pure reactive navigation, but to motion planning approaches in the line of Rapidly-exploring

Random Trees [LaV01]. However, this topic has been not explored in this thesis.

16.4.2 C PTG: Circular trajectories

This is the simplest path model, and the only one used in previous works on reactive navigation. Velocities remain constant with time along a given trajectory, as follows from the design equations in Figure 16.7. In this case Eq. (16.3.2) is integrable and gives us the following closed form expression for trajectories:

$$\mathbf{q}(\alpha,t)=\begin{cases}\begin{bmatrix}\frac{Kv_0}{\omega_0\tan\frac{\alpha}{2}}\sin\left(t\omega_0\tan\frac{\alpha}{2}\right)\\ \frac{Kv_0}{\omega_0\tan\frac{\alpha}{2}}\left(1-\cos\left(t\omega_0\tan\frac{\alpha}{2}\right)\right)\\ t\omega_0\tan\frac{\alpha}{2}\end{bmatrix} & ,\alpha\neq 0\\ \begin{bmatrix}Kv_0t\\0\\0\end{bmatrix} & ,\alpha=0\end{cases} \tag{16.4.1}$$

where the parameter K is introduced for accounting for forward and backward trajectories (for values of 1 and -1, respectively).

16.4.3 α-A PTG: Trajectories with asymptotical heading

These trajectories are generated by linear/angular velocities which are directly/inversely proportional to the difference between the robot heading and the parameter α. As a result, trajectories tend asymptotically to straight paths from the origin (see the corresponding diagram in Figure 16.7). We have verified that for our robots moving in office-like scenarios this is one of the most frequently selected PTG. Unfortunately, this PTG results in non-integrable trajectories and requires numerical solutions.

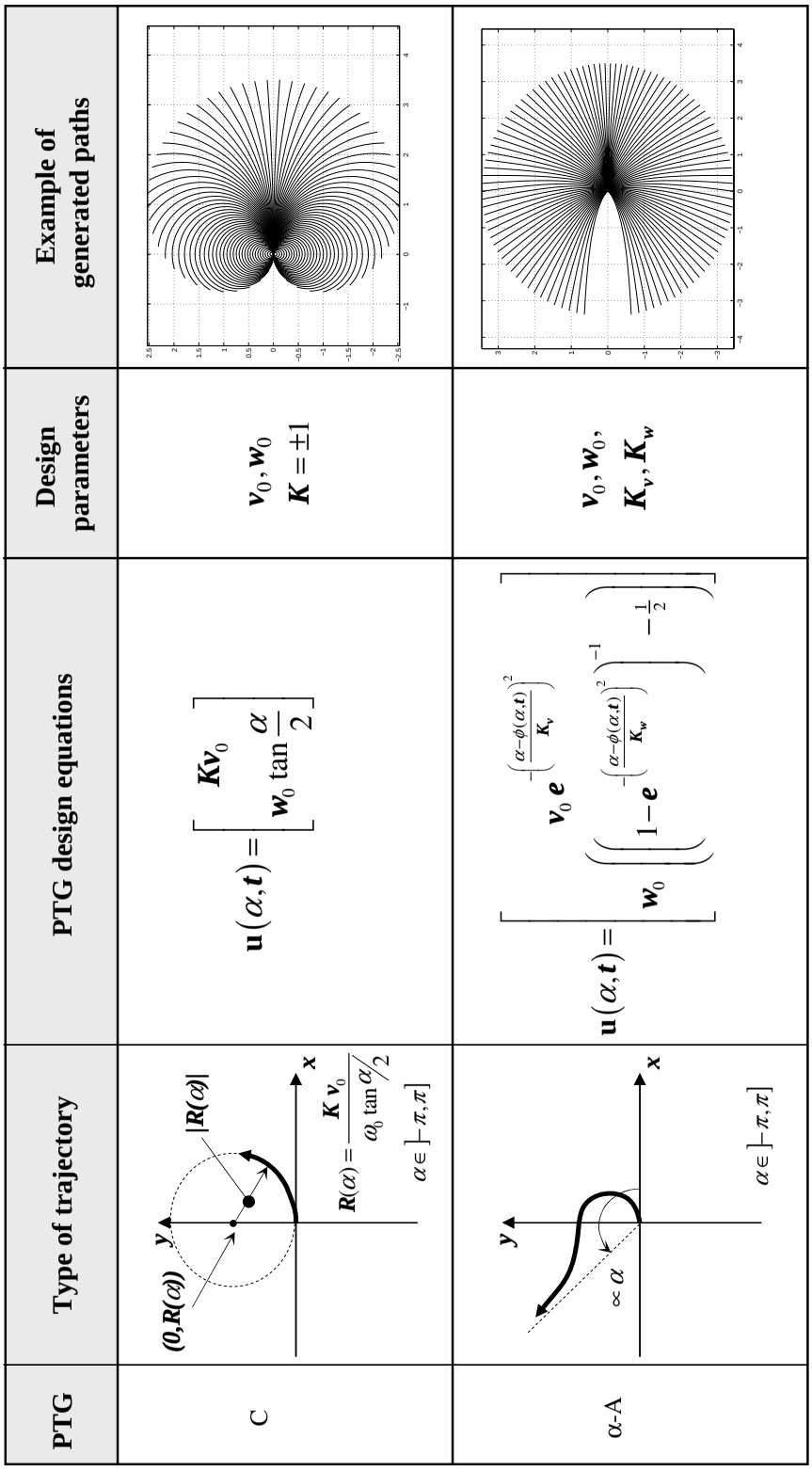


Figure 16.7: The table shows the relationship of the parameter α with the type of trajectory, the design equations (the trajectory velocity vector), and a graphical example of generated paths in C-Space (as a 2-d top view).

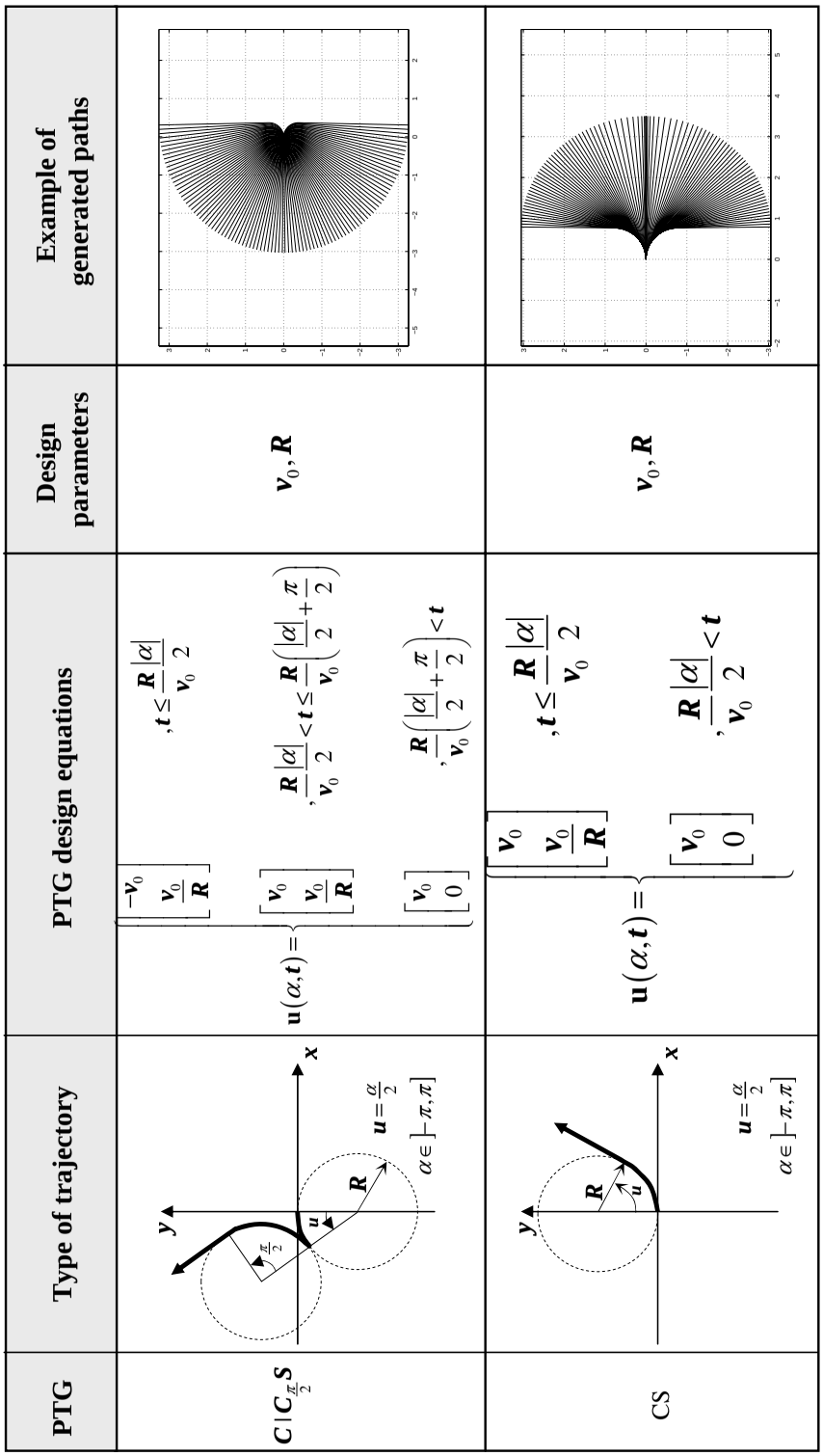


Figure 16.8: The table shows the relationship of the parameter α with the type of trajectory, the design equations (the trajectory velocity vector), and a graphical example of generated paths in C-Space (as a 2-d top view).

16.4.4 C|C$_{\frac{\pi}{2}}$S and CS PTG: A set of optimal paths

These path models have been well studied in the field of motion planning, and represent some of the optimal (shortest) path models for a car-like robot with a minimum turning radius [Ven99]. To the best of our knowledge, the present approach is the first one to enable the incorporation of optimal solutions from the motion planning field into a reactive method. In this case both models lead to integrable expressions, although they are omitted for clarity since the results are straightforward concatenations of circular and straight segments.

16.5 Mapping the real environment into TP-Space

Navigating into a TP-Space implies two transformations: (i) the construction of TP-Obstacles, and (ii) the translation of the target position (without orientation) into TP-Space. Although both transformations map 2-d points from the environment into TP-Space, they are managed in quite different ways. The target location remains as a single point in TP-Space and it is computed at each iteration by the inverse PTG function, ignoring the robot heading at it. Regarding the translation of obstacles, we will assume without loss of generality that only point obstacles exist. If any other kind of obstacles needs to be modeled (e.g. polygonal obstacles) they can be parameterized as a continuous sequence of point obstacles and TP-Obstacles built by composition. When point obstacles are moved into TP-Space, each point becomes a region, named TP-Obstacle. Let $o \in \mathbb{R}^2$ be a real obstacle point in WS, $C - Obstacle(o)$ its representation in C-Space, and Surf(PTG) the C-Space surface of a PTG. We define the TP-Obstacle for point o as:

$$\text{TP-Obstacle}(o) = \left\{(\alpha, d) | (\alpha, d) = PTG^{-1}(\mathbf{q}), \forall \mathbf{q} \in \text{C-Obstacle}(o) \cap \text{Surf(PTG)}\right\}$$

That is, TP-Obstacles are defined as the transformation into TP-Space (by means of the inverse PTG mapping) of the intersection between C-Obstacles and the sampling surface of the PTG. Since a robot can collide with any obstacle from many different poses, TP-Obstacles are always two-dimensional regions in TP-Space. Holonomic obstacle avoidance methods will measure obstacle distances along straight paths in TP-Space, thus, in practice, only the closest obstacle must be kept for each direction of the transformed space. The whole process of transforming obstacles into TP-Space is illustrated in Figure 16.4(c), where the rightmost graph shows a polar plot of the closest TP-Obstacle at each direction as normalized distances. TP-Space charts from now on are shown using this polar representation.

The process of building TP-Obstacles implies computing only the part of C-Obstacles that contributes with useful information: the intersection with a given sampling surface. Although this is a computational expensive operation, an efficient method to obtain an approximate solution has been developed. It must be remarked that exact solutions may exist for integrable PTGs, but we choose to rely on an approximate method to enable the utilization of any PTG regardless its integrability. The method for computing the intersection of C-Obstacles with the PTG sampling surface is based on a lookup table with collision tests precomputed from simulations. A closely related idea was previously reported in [Sch98] for the case of circular paths and any-shape robots, although here the process is extended to an arbitrary path model given by a PTG: the physical space around the robot is arranged into a rectangular grid whose cells store their associated TP-Obstacle, built from the set of pairs (α, d) that lead to collision, as illustrated in Figure 16.9. Therefore, the problem of translating a set

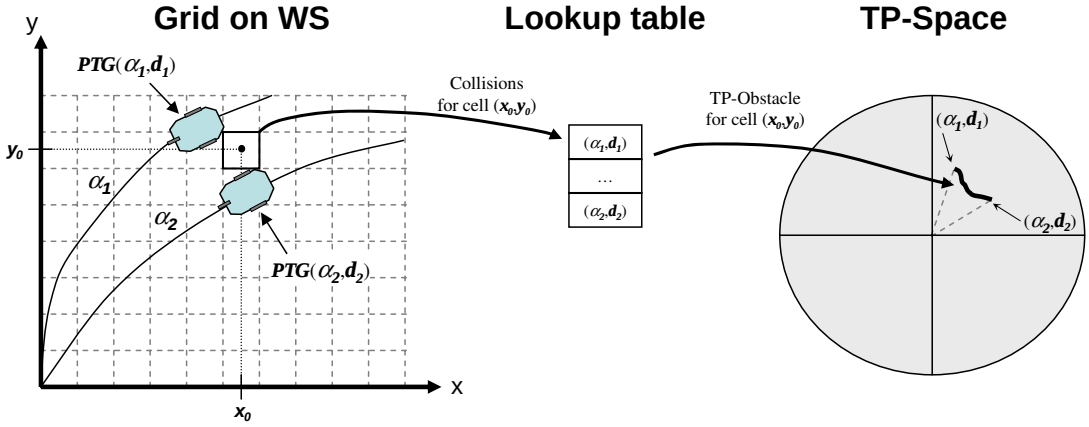


Figure 16.9: The process for a fast construction of TP-Obstacles involves building a rectangular grid where each cell keeps the collision pairs (α, d) for its associated TP-Obstacle and a given PTG. Then moving real obstacles into TP-Space is accomplished by the addition of the occupied cells.

of obstacles into TP-Space becomes that of adding the TP-Obstacles elements for the occupied cells.

16.6 From movement commands to real world actions

We consider now that the motion generated by the holonomic collision avoidance method in the virtual WS (the TP-Space) is a pair stating the robot desired speed s and direction α_m (in the interval $]-\pi, \pi]$). This motion command is translated back into a real robot movement in two steps:

1. We obtain a normalized velocity command as

$$U_{norm}(\alpha_m) = \mathbf{u}(\alpha_m, 0) = [v(\alpha_m, 0)\omega(\alpha_m, 0)]^\top$$

i.e. through the evaluation of the PTG design functions at $\alpha = \alpha_m$. We take the initial response (at $t = 0$) since the PTG reference system is the robot current pose, i.e. the robot is always at the origin of trajectories in the TP-Space.

2. The velocity command for the real robot $\mathbf{u}_{rob}$ is computed by scaling $\mathbf{u}_{norm}$ according to the holonomic velocity s in TP-Space (provided by the holonomic method). Mathematically:

$$\mathbf{u}_{rob}\left(\alpha_m, s\right) = s \cdot \frac{\mathbf{u}_{norm}\left(\alpha_m\right)}{\max_\alpha m\left(\mathbf{u}(\alpha, 0)\right)} \tag{16.6.1}$$

where $m(\cdot)$ is the custom metric defined in [Bla08f], which in this case gives us:

$$\begin{aligned} m\left(\mathbf{u}\left(\alpha,0\right)\right) &= m\left(\frac{\partial P\left(\alpha,\tau\right)}{\partial \tau}\right) \\ &= \left\| \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & r \end{bmatrix} \begin{bmatrix} \cos\phi\left(\alpha,0\right) & 0 \\ \sin\phi\left(\alpha,0\right) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v\left(\alpha,0\right) \\ \omega\left(\alpha,0\right) \end{bmatrix} \right\| \\ &= \sqrt{v(\alpha,0)^2 + (r \cdot \omega(\alpha,0))^2} \end{aligned} \tag{16.6.2}$$

16.7 A complete reactive navigation system

In order to implement a navigation based on TP-Space, it is presented next a complete navigation system, sketched in Figure 16.10, which is a detailed view of Figure 16.3(b). The overall system comprises a control loop where sensor readings along with an estimation of the target relative location are supplied to the reactive navigation system.

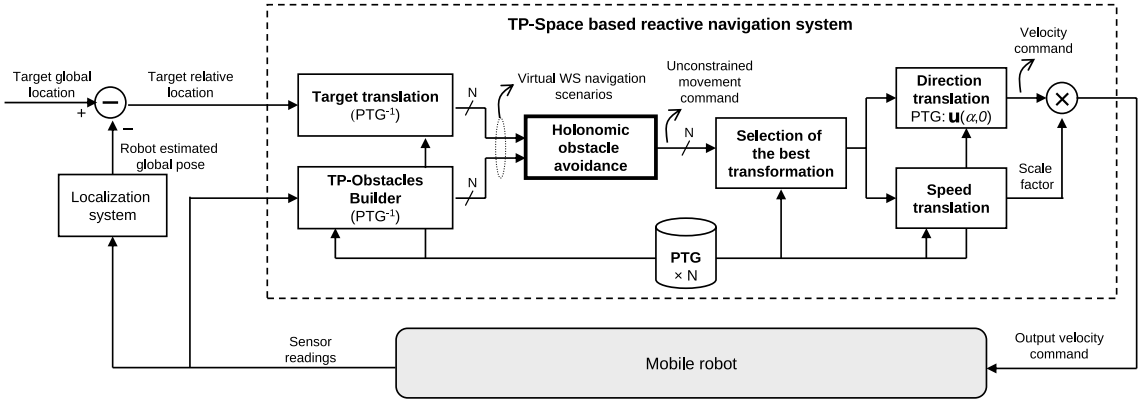


Figure 16.10: A complete reactive navigation system based on TP-Space involves translating obstacles and the target location into the TP-Space, through a variety of PTGs simultaneously. Each one generates a virtual WS navigation scenario which is solved by a simple obstacle avoidance method. Next, the resulting movements are evaluated to find the most advantageous transformation, which is selected to generate the real robot velocity command using the PTG design equations.

This system computes a velocity command that is sent back to the robot, closing the control loop. This process is repeated periodically (e.g. at 10Hz), resulting in a fast response to dynamic obstacles while steering the robot towards the target. We assume that the target location is given in some fixed coordinate system, and that a localization system (not addressed here) is available to accurately estimate the position of the target relative to the robot.

Within the system, firstly the sensed obstacles and the target are translated into TP-Space, typically using several PTGs simultaneously, yielding a different virtual WS (TP-Space) for each PTG. Next, a holonomic obstacle avoidance method is applied to each one to end up with their respective movement commands. To select the most advantageous one, we evaluate them by weighting the following factors (normalized within the range [0,1]):

- f1: Collision-free distance for the selected movement direction (in TP-Space).
- f2: Direction relative to target (in TP-Space). Let α_m be the desired movement

direction, and α_t the direction towards the target, in the transformed space. Then:

$$f_2 = \exp\left\{-\left(\frac{\alpha_m - \alpha_t}{2\pi/3}\right)^2\right\} \tag{16.7.1}$$

- f3: Robot heading towards target (in C-Space). This term prioritizes the movements that make the robot to face the target, in the real world. If θ is the heading of target at the end of robot direction movement, then this factor can be evaluated as:

$$f_3 = \exp\left\{-\left(\frac{\theta}{\pi/2}\right)^2\right\} \tag{16.7.2}$$

- f4: Euclidean distance to the target (in WS). This factor gives more relevance to movements that take the robot closer to the target:

$$f_4 = \frac{d_{max} - \min\{d_t, R_m\}}{R_m} \tag{16.7.3}$$

where d_t is the minimum distance from robot to target in the selected trajectory and R_m is the normalization factor specified in the PTG design.

- f5: Hysteresis. This factor contributes to stabilize the behavior of the system, avoiding high-frequency oscillations between different possible transformations that perform similarly through short periods of time:

$$f_5 = \begin{cases} 1, \text{ if this PTG was selected in the last iteration.} \\ 0, \text{ otherwise.} \end{cases} \tag{16.7.4}$$

Logically, the global behavior of the system depends on the values of these factors, though empirically we have determined that a rough tuning suffices to achieve a good performance in most situations and on different robots. In our implementation we

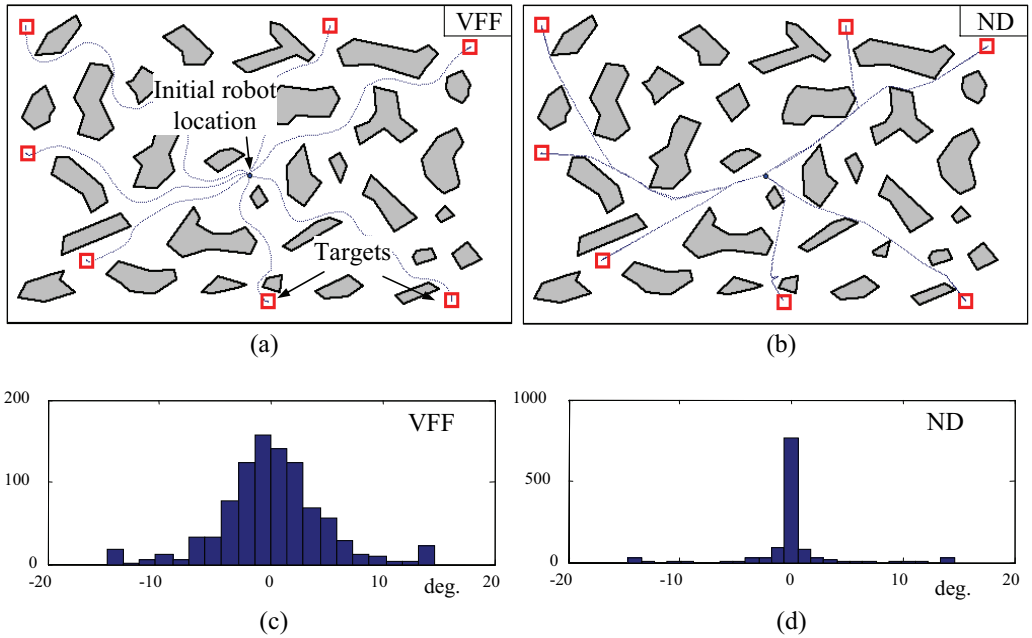


Figure 16.11: A comparison between two implemented holonomic obstacle avoidance algorithms in a synthetic environment: (a) and (b) show navigations from a central point towards seven different target locations using the Virtual Force Field (VFF) and the Nearness diagram (ND) methods, respectively. The later generates straighter paths for the virtual holonomic robot, which makes the real robot less shaky in cluttered environments. This is confirmed by the histograms of the changes in the steering between iterations, shown in (c) and (d) for both methods.

have applied the weights 0.4, 0.15, 0.15, 0.2, 0.1 for each one, respectively. Intuitively, this distribution of the weights gives a maximum relevance to the clearance (term f_1), and in second place to the closeness of the trajectory toward the target (term f_4). The transformation with the highest score is chosen at each iteration, and then its corresponding movement is converted back into a velocity command for the real robot as discussed in a previous section: first, a normalized command is directly extracted from the PTG design equations, and then it is scaled to account for the speed given by the obstacle avoidance method.

Two holonomic obstacle avoidance methods have been tested with TP-Space: a Virtual Force Field (VFF) method and a custom implementation of the Nearness Diagram

approach [Min04]. To compare the performance of these methods on a kinematically free robot, some simulated experiments have been carried out, whose results are shown in Figure 16.11. A major distinctive feature of ND is that it results in mostly straight paths, which is reflected in few changes in the real robot direction (see the histograms of Figure 16.11(c)-(d)). Nevertheless, VFF is simpler to implement than ND, and performs equally well in uncluttered environments.

The major concern in getting a real-time functional implementation of this navigation system is to have a fast TP-Obstacles building process, which has been achieved through the abovementioned efficient procedure based on precomputed lookup-tables. One lookup-table is stored for each PTG for a fast consultation during navigation. Although building these tables takes a long time (typically 15-35 seconds for our configuration), they must be updated only if the shape of the robot changes; on the other hand, they allow TP-Obstacles to be built in such a short time as 0.2ms per PTG. This quick transformation time and the also quick response of the holonomic methods keep the whole system fast enough to perform in real-time within dynamic scenarios.

16.8 Experimental evaluation and discussion

PTG-based reactive navigation has been intensively tested for more than two years in office-like scenarios with our robots SENA, a robotic wheelchair [Gon06a,Gon06b], and Sancho [Gon09b], a service robot built upon a Pioneer 3-DX mobile base, both driven differentially and equipped with laser range finders for obstacle detection. Next we discuss four different experiments, each one aimed to illustrate a differentiated feature of the proposed navigation system.

16.8.1 First experiment: simulated robot

This experiment demonstrates the feasibility of our approach for dealing with a car-like robot in simulated cluttered scenarios. It must be remarked that any reactive method that ignores the robot kinematics will fail if applied to a kinematically-constrained robot. The car-like robot is commanded to navigate towards the target point shown in Figure 16.12(a) making use of two PTGs: $C|C_{\frac{\pi}{2}}S$ and CS, both being able to generate (possibly) optimal paths for this kind of robot [Ven99]. The selection of the active PTG at each instant of time is graphically represented in Figure 16.12(b), where it can be seen how the $C|C_{\frac{\pi}{2}}S$ model is solely used when maneuvering to get the robot out of the starting location. Notice how our approach allows such a maneuvering in a memoryless system. Nevertheless, it must be stressed that any reactive system, including the one described in this work, has limited foreseeing capabilities and may not be able to find a valid path in all the situations.

16.8.2 Second experiment: real robotic wheelchair

In this experiment with the robotic wheelchair SENA [Gon06a] we illustrate the feasibility of controlling a kinematically constrained real mobile platform with a strong rectangular-like shape using a simple obstacle avoidance method (ND in this case). Ignoring the shape of this robot would most probably lead to collision. The resulting trajectory for this experiment is shown in Figure 16.13(d), and some snapshots during the navigation can be seen in Figure 16.13(e)-(h). Here we have employed four PTGs simultaneously: a forward C PTG, a backward C PTG, and two different instances of the α-A PTG, whose selection over time is plotted in Figure 16.13(a). In this experiment the backward movement (PTG with index 1) is never selected.

Regarding the time requirements of the system for this configuration of PTGs, it

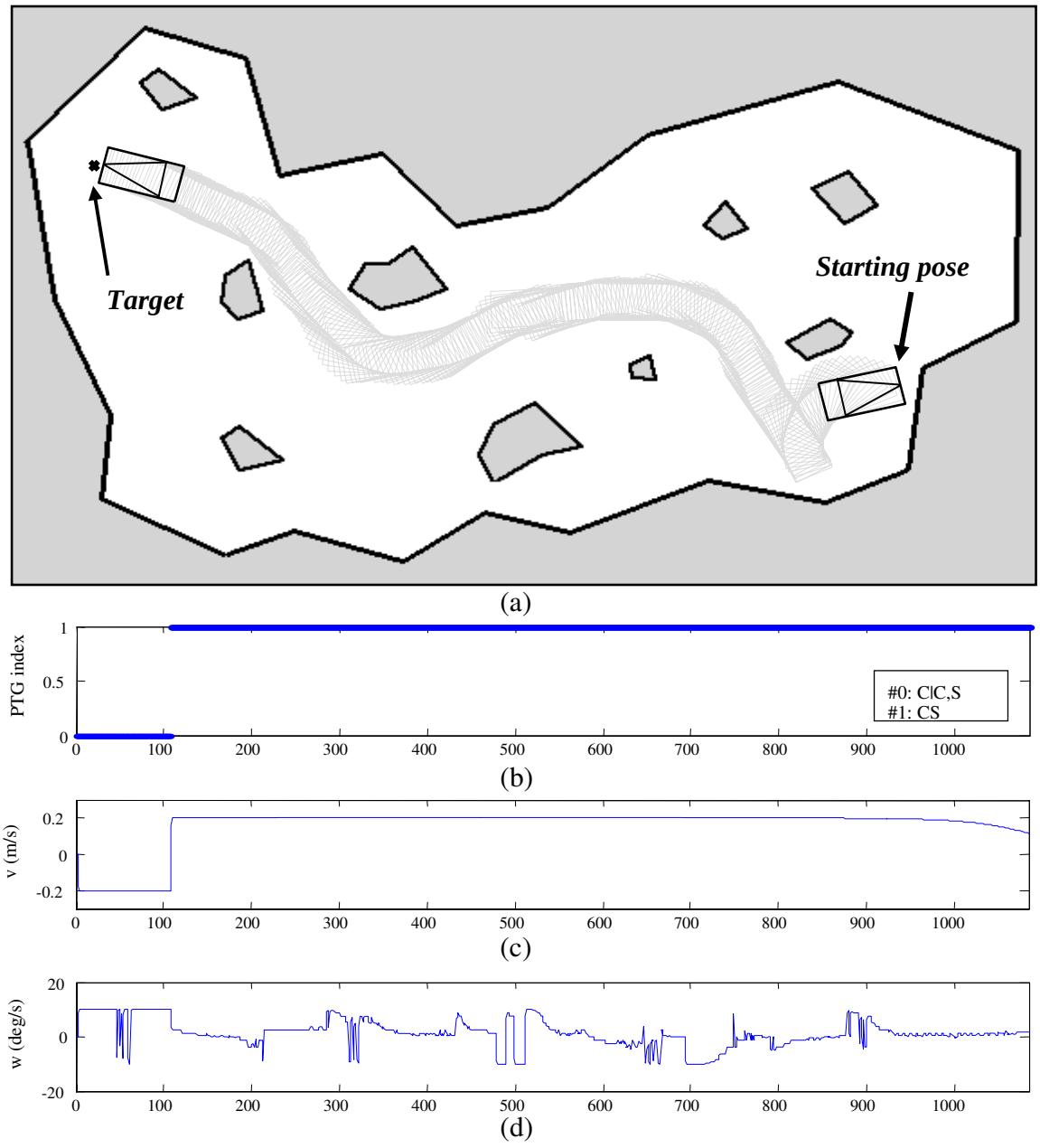


Figure 16.12: A simulated experiment to demonstrate the manoeuvring capabilities of a car-like robot with our reactive navigation system. (a) The resulting trajectory, (b) the active PTG at each instant of time, and (c)-(d) the linear and angular velocities of the robot.

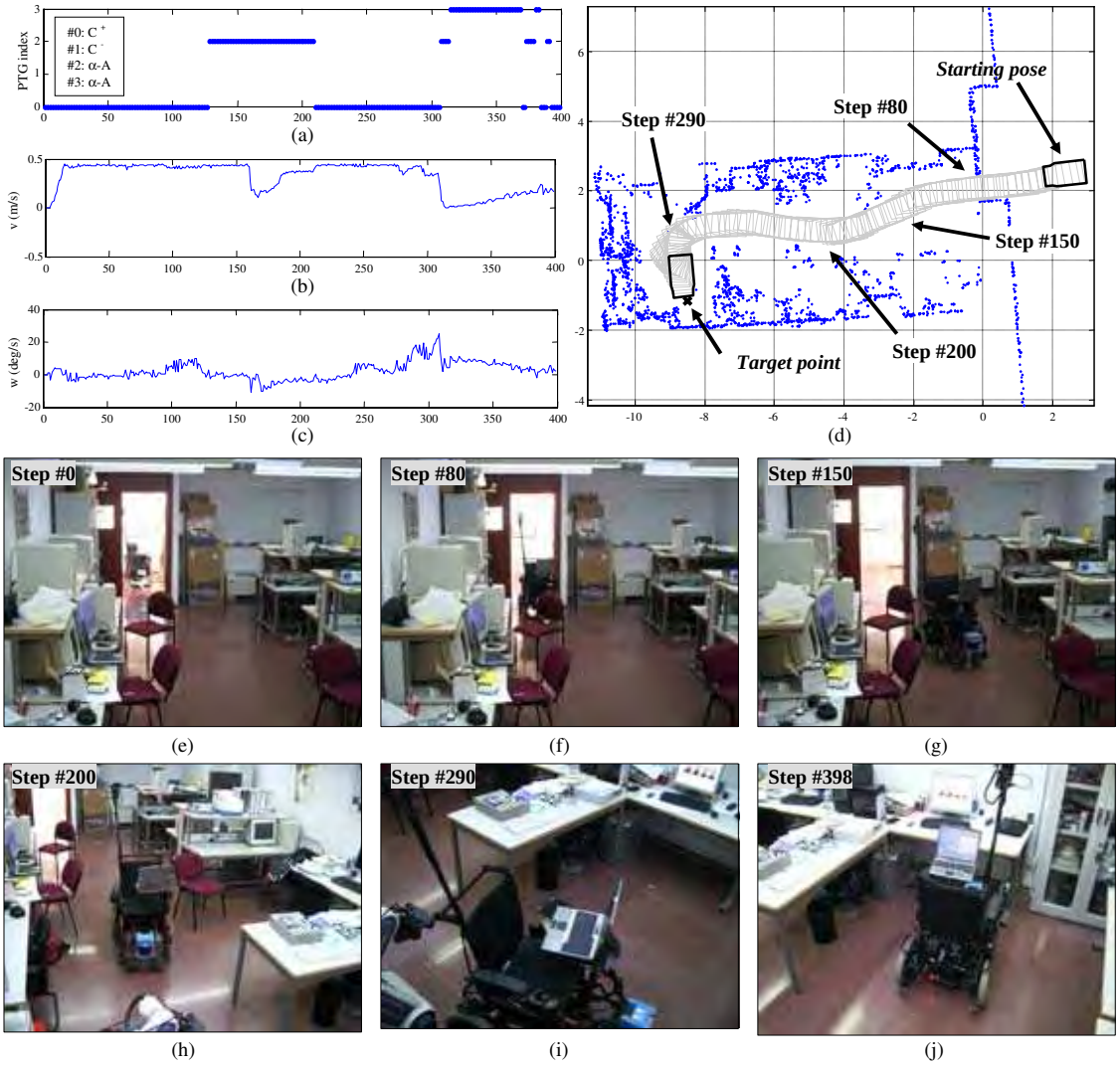


Figure 16.13: In this experiment, SENA (a robotic wheelchair) performs a reactive navigation using four PTGs, whose selection over time is shown in (a). The linear and angular velocities along the navigation are plotted in (b)-(c), respectively, and the resulting trajectory is depicted in (d). Snapshots (e)-(j) show some instants along the course of the navigation

takes a mean of 1.22ms (on a 1.8GHz Pentium-M) to execute one complete step of the navigation system, including the mapping of all the obstacles into TP-Space, ND-based collision avoidance, the selection of the best movement, and its transformation into a command for the real robot. Provided that the execution frequency is 10Hz, our method occupies a mere 1.22% of the CPU time.

16.8.3 Third experiment: comparison to the "arc approach"

It is worth to illustrate a situation where our approach could be compared to previous reactive techniques in order to reveal its advantages. The following experiment consists of our mobile robot Sancho, equipped with a front and a rear laser scanner, being commanded a reactive navigation from a cluttered room towards a corridor outside. The navigation is repeated twice: the first time the robot uses a set of five PTGs (of types C, α-A, and CS), while the second time it uses just forward and backward circular arcs (type C PTG). The overall trajectories of the robot for each case are shown in the two graphs at the top of Figure 16.14. In this cluttered environment the obstacle avoidance method (the ND algorithm) has problems finding a good movement by using just circular arcs. One can clearly observe a much poorer performance of the navigation in this case, in contrast to the case of the five PTGs, as also revealed by the overall path length and time taken by each navigation, plotted in Figure 16.14.

It is remarkable that by using more path models (five PTGs) the robot accomplishes the navigation in 88 seconds less than using just circular arcs, also saving 11.4m from the overall path length. One of the reasons of these differences is that, for the case of the circular arcs, the robot encountered significant problems at some specific places along its trajectory, such as turning a pronounced corner of about 90º or passing the doorway by the end of the navigation. For comparison purposes, both events are marked (with

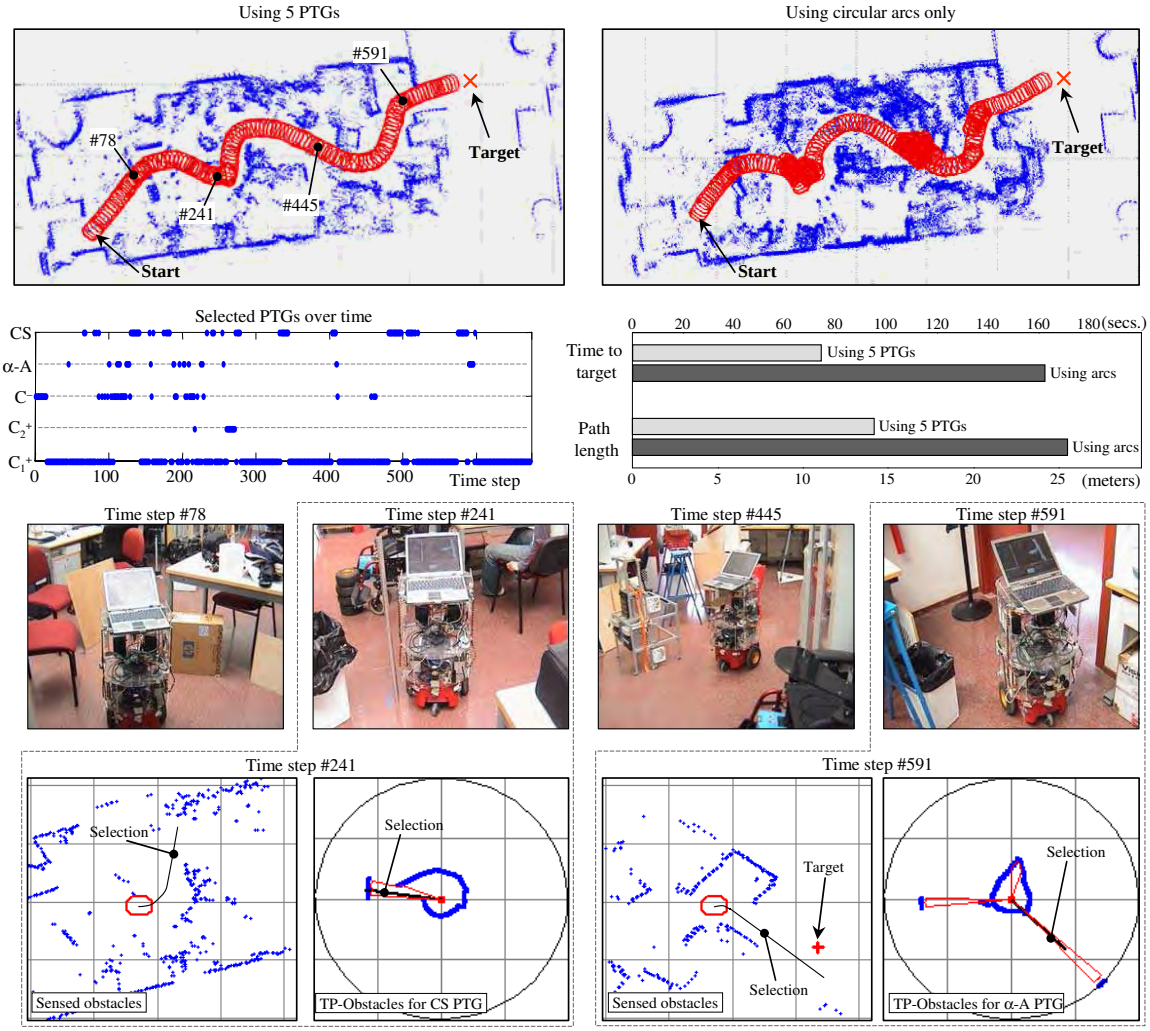


Figure 16.14: A comparison of performance between the same obstacle avoidance method using five PTGs and only circular (forward and backward) arcs. It is shown how the use of several PTGs improves both the time consumed by the navigation and the overall path length. Some snapshots along the navigation show the cluttered scenario and the sensed obstacles at some specific instants of time, both in the robot workspace and in the TP-Space using the PTG selected at each time. See the text for further discussion.

their respective time steps 241 and 591) in Figure 16.14 for the path described using five PTGs. To illustrate how the additional PTGs have improved the navigation of the robot at those specific parts of the navigation, in that figure we show the sensed obstacles and their translations into TP-Space for the PTGs selected at those time steps. At time step 241, a PTG of the type CS (refer to §16.4.4) is selected since it includes a movement which fits perfectly to the maneuver the robot needs to bypass the obstacle. The narrow "gap" in the TP-Obstacles which appears in this case represents the small range of CS trajectories (α parameter values) that makes the robot to pass by the corner without colliding. Similarly, the graphs for time step 591 show how a PTG of the type α-A reveals collision-free maneuvers for passing through the narrow doorway. We must emphasize that these PTGs are selected at these specific instants of time because the circular arcs are not able to find better movements using the criterion of the obstacle avoidance algorithm.

To sum up, this experiment has described an example of how the use of other path models apart from arcs introduces a significant improvement into the overall performance of the navigation. Despite the fact that circular arcs may be the optimal choice for the most part of a typical navigation, the existence of other choices at some specific moments allows the robot to find maneuvers that would be otherwise much more difficult (or even impossible) to find by means of arcs, a fact that ultimately may determine the performance and reliability of the whole reactive navigator.

16.8.4 Fourth experiment: dynamic environments

Finally, we present an experiment where SENA, the robotic wheelchair, navigates for almost two hours in a non-prepared, crowded scenario. We have chosen for this experiment the entrance of our building at the University of Málaga. The robot is commanded

to navigate repeatedly between two fixed target locations, labeled as 'A' and 'B' in Figure 16.15(a)-(b), and in its way it must avoid some static obstacles (columns, furniture, etc.) and moving people.

During the experiment the robot avoids collisions with several people walking in the opposite direction than the robot, as well as some students who intentionally try to block its way. In all the cases, the reactive navigator (using five PTGs in this case) successfully avoids all the collisions and gets out of the blocking situations.

To perform this experiment in such a dynamic scenario, we have required robust localization within the environment, which has been carried out through particle filter-based localization (see Chapter 4) for occupancy grid maps (see Figure 16.15(a)). For robustness both in localization and in detecting all the surrounding obstacles, we instrumented the robotic wheelchair with three laser range scanners at different heights and positions: one at the front, a rear scanner, and another one on the top of the robot, as highlighted in Figure 16.15(e).

16.8.5 Discussion

We have presented an approach for the reactive navigation of a kinematically-constrained and any-shape mobile robot, on the basis of a clear and useful separation of the problems of robot shape and kinematic restrictions, and collision avoidance. For that, a generalized kinematics abstraction mechanism has been developed which allows us using a variety of path models (PTGs) to obtain a better sampling of the whole C-Space from which more and better collision-free paths towards the target location can be found. The generalization of path models enables the introduction of optimal path models into our reactive navigation system for the first time in a non-planned approach to robot navigation. The implementation of the navigation system has demonstrated

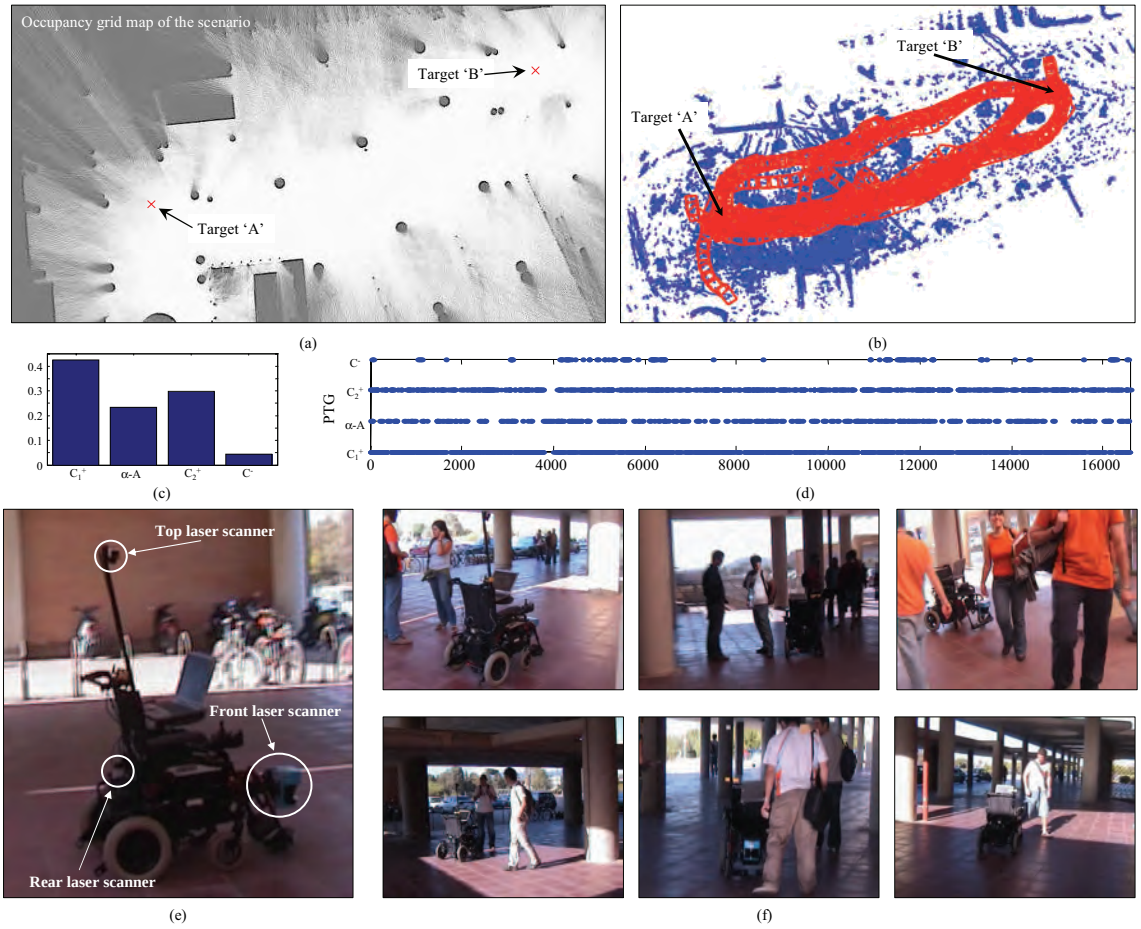


Figure 16.15: In this experiment SENA navigated for two hours in a non-prepared scenario amidst dozens of students without colliding. (a) An occupancy grid map of the scenario. (b) Overlap of laser scans during the mission, which consists of going to a pair of targets sequentially. (c) Ratio of selection for each PTG, and (d) their selection over time. (e) Positions of the three laser scanners on the robot. (f) Some snapshots of the experiment.

to be effective and very efficient for robots in cluttered, dynamics scenarios, which has been verified along more than a year of extensive tests. PTG-based navigation in non-pure reactive approaches is an interesting research line that will be addressed in the future.

Appendices

APPENDIX A

GEOMETRY CONVENTIONS

A.1 Conventions

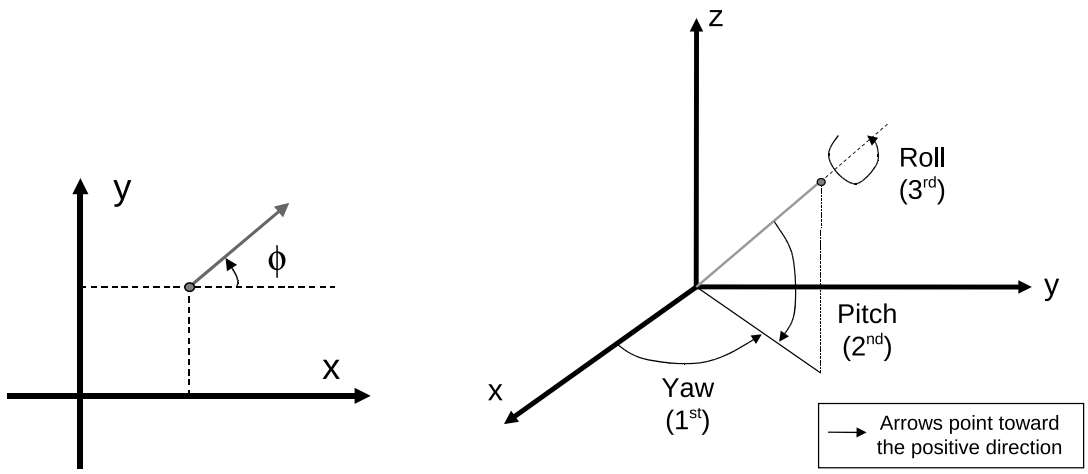
In this section we review the conventions employed in this thesis for designating geometric positions with orientations, that is, *poses*.

For the case of a 2-d pose, it comprises a 2-d location (x, y) plus a single heading angle ϕ. The positive direction of this angle is illustrated in Figure A.1(a).

On the other hand, 3-d poses are represented by a 3-d location (x, y, z) and three rotation angles. In this case we follow the general convention in which the order of the rotations is around the axes z, y and x, respectively. Each rotation is applied following the so-far transformed axes, not the original ones. This process can be observed in Figure A.1(b).

A.2 Operations

The most important operations with poses are composition and inverse composition. The former consists of concatenating two poses or a pose and a point. Given a point



(a) 2-d pose (b) 3-d pose

Figure A.1: Representation of 2-d and 3-d poses and the direction of positive angles in each case.

p and a pose q, their composition is denoted as [Lu97a, Smi88]

$$p' = p \oplus q$$

and its geometrical meaning is that the transformed point p' is obtained by moving the original point p to the new system of reference defined by q. The inverse composition can be used to reverse this operation, that is, to compute the coordinates of the point p' "as seen from" the reference system of q. This operation is denoted as

$$p = p' \ominus q$$

Both operations can be implemented by means of homogeneous coordinates, where 2-d or 3-d points are represented by 3 o 4-vectors where the extra element is fixed to the unit, that is,

$$X = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad \text{, or} \quad X = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Transforming a point is accomplished by means of left-multiplying its vector by the homogeneous matrix of the pose for the case of composition, or by its inverse homogeneous matrix for the case of inverse composition.

The matrix of a 2-d pose q is given by

$$R_q = \begin{bmatrix} \cos\phi & -\sin\phi & x \\ \sin\phi & \cos\phi & y \\ 0 & 0 & 1 \end{bmatrix}$$

and that for a 3-d pose can be obtained by left-multiplying the three rotations around the z, y and x axes, in that order, giving us:

$$\begin{aligned} R_q &= R_z \cdot R_y \cdot R_x \\ &= \begin{bmatrix} \cos\phi\cos\chi & \cos\phi\sin\chi\sin\psi - \sin\phi\cos\psi & \cos\phi\sin\chi\cos\psi + \sin\phi\sin\psi & x \\ \sin\phi\cos\chi & \sin\phi\sin\chi\sin\psi + \cos\phi\cos\psi & \sin\phi\sin\chi\cos\psi - \cos\phi\sin\psi & y \\ -\sin\chi & \cos\chi\sin\psi & \cos\chi\cos\psi & z \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned}$$

where the yaw, pitch and roll angles are denoted as ϕ, χ and ψ, respectively.

APPENDIX B

NUMERICALLY-STABLE METHODS FOR LOG-LIKELIHOODS

This appendix describes practical methods to overcome problems with the numerical stability of operations with logarithmic weights or likelihood values. Logarithmic scales are widely employed in particle filter implementations to greatly extend the dynamic range of particle weights.

B.1 Average

Let $\{ll_i\}_{i=1}^N$ be a set of N log-likelihoods such as $ll_i = \log(l_i)$. We are interested in the log-average of all the l_i, denoted as $\log(\bar{l})$, where

$$\bar{l} = \frac{1}{N} \sum_{i=1}^{N} l_i.$$

Operating,

$$\log \bar{l} = \log \left(\frac{1}{N} \sum_{i=1}^{N} l_i \right) = -\log N + \log \left(\sum_{i=1}^{N} \exp\left(ll_i\right) \right)$$

The numerical problem comes from the operation $\exp(ll_i)$, where overflows could easily occur. To solve this, we define the maximum log-likelihood $ll_{max} = \max_i \ ll_i$ and rewrite the equation above as

$$\begin{aligned} \log \bar{l} &= -\log N + \log \left(\exp\left(ll_{max}\right) \sum_{i=1}^{N} \exp\left(ll_i - ll_{max}\right) \right) \\ &= -\log N + ll_{max} + \log \left(\sum_{i=1}^{N} \exp\left(ll_i - ll_{max}\right) \right) \end{aligned}$$

which is the desired numerically-stable expression, since the arguments of the exponential are biased such as the maximum value becomes $\exp(ll_{max} - ll_{max}) = 1$.

B.2 Weighted average

In this case, we are interested in a weighted average $\bar{l}$ of likelihoods l_i with weights w_i, that is,

$$\bar{l} = \frac{\sum_{i=1}^{N} l_i \cdot w_i}{\sum_{i=1}^{N} w_i}$$

where both the weights and the likelihoods are given in a logarithmic scale:

$$\begin{aligned} ll_i &= \log l_i \\ lw_i &= \log w_i \end{aligned}$$

Operating with $\log(\bar{l})$ we obtain

$$\log \bar{l} = \log \left(\frac{\sum_{i=1}^{N} \exp(ll_i) \exp(lw_i)}{\sum_{i=1}^{N} \exp(lw_i)} \right) = -\log \left(\sum_{i=1}^{N} \exp(lw_i) \right) + \log \left(\sum_{i=1}^{N} \exp(ll_i + lw_i) \right)$$

where, as in the previous section, the numerical problems arise from potentially too large (or too small) arguments in the exponentials. If we define the maximum of the log-weights and log-likelihoods as

$$\begin{aligned} lw_{max} &= \max_i \ lw_i \\ ll_{max} &= \max_i \ ll_i \end{aligned}$$

, respectively, the equation above becomes

$$\begin{aligned} \log \bar{l} \ = \ & -\log \left(\sum_{i=1}^{N} \exp(lw_i - lw_{max}) \right) \\ & + \log \left(\sum_{i=1}^{N} \exp(ll_i + lw_i - ll_{max} - lw_{max}) \right) \\ & + ll_{max}. \end{aligned}$$

APPENDIX C

BERNOULLI TRIALS IN REJECTION SAMPLING

Consider two probability distributions, which we will call *prior* and *observation likelihood*, to be normally distributed according to the one-dimensional densities $\mathcal{N}(0, \sigma_p^2)$ and $\mathcal{N}(\Delta\mu, \sigma_o^2)$, respectively. For convenience we assume the prior to be centered at zero, with $\Delta\mu$ modeling the separation between the two distributions.

In the following we derive the probability p of accepting one random sample in a rejection sampling trial, where a sample is drawn from the prior and is accepted with a probability proportional to the associated observation likelihood, that is:

$$p = \frac{p(z|x)}{\max\{p(z|x)\}} \tag{C.0.1}$$

where $p(z|x)$ denotes the observation likelihood. Since $p(z|x)$ is a Gaussian, the quotient above reduces to:

$$p = \exp\left\{-\frac{1}{2}\left(\frac{x - \Delta\mu}{\sigma_o}\right)^2\right\} \tag{C.0.2}$$

Observe how this value depends on the sample x, which is distributed following the prior $\mathcal{N}(0, \sigma_p^2)$. Thus, we can estimate the expected value of p by:

$$E[p] = \int_{-\infty}^{\infty} \exp\left\{-\frac{1}{2}\left(\frac{x-\Delta\mu}{\sigma_o}\right)^2\right\}\mathcal{N}(x; 0, \sigma_p^2)dx \tag{C.0.3}$$

By multiplying and dividing by the appropriate constant, the terms inside the integral can be transformed into the product of two Gaussians:

$$E[p] = \sigma_o\sqrt{2\pi}\int_{-\infty}^{\infty} \mathcal{N}(x; \Delta\mu, \sigma_o^2)\mathcal{N}(x; 0, \sigma_p^2)dx \tag{C.0.4}$$

Following a probabilistic interpretation of the integral above similar to that discussed in §2.5, it can be shown that Eq. (C.0.4) can be rewritten as

$$\begin{aligned} E[p] &= \sigma_o\sqrt{2\pi}\cdot\mathcal{N}(0; \Delta\mu, \sigma_o^2 + \sigma_p^2) \\ &= \frac{\sigma_o\sqrt{2\pi}}{\sqrt{2\pi(\sigma_o^2+\sigma_p^2)}}\exp\left(-\frac{1}{2}\frac{\Delta\mu^2}{\sigma_o^2+\sigma_p^2}\right) \end{aligned}$$

For convenience, we define the constant $\tau = \sigma_o/\sigma_p$, such as:

$$E[p] = \frac{1}{\sqrt{1+\tau^{-2}}}\exp\left(-\frac{\Delta\mu^2}{2\sigma_p^2(1+\tau^2)}\right)$$

which is the final expression used for the numerical results shown in §4.4.

APPENDIX D

MAXIMUM MEAN INFORMATION FOR A SYNTHETIC ENVIRONMENT

As the resolution of an occupancy grid increases, its mean information (MI), as defined in §10.4, tends towards a maximum value. Unfortunately, a closed form expression for this bound can not be derived in the general case since it depends on the environment, the inverse sensor model and the number of observations merged into the map. For a particular case, however, we can derive such an expression to demonstrate the convergence of the MI and its high independence of the grid resolution.

Assume a robot equipped with a 360° field-of-view radial range sensor, standing at the center of a circular environment of radius R, and making only one observation (z_1). Due to the circular symmetry we consider the circular grid map $p(m_\rho)$ instead of the

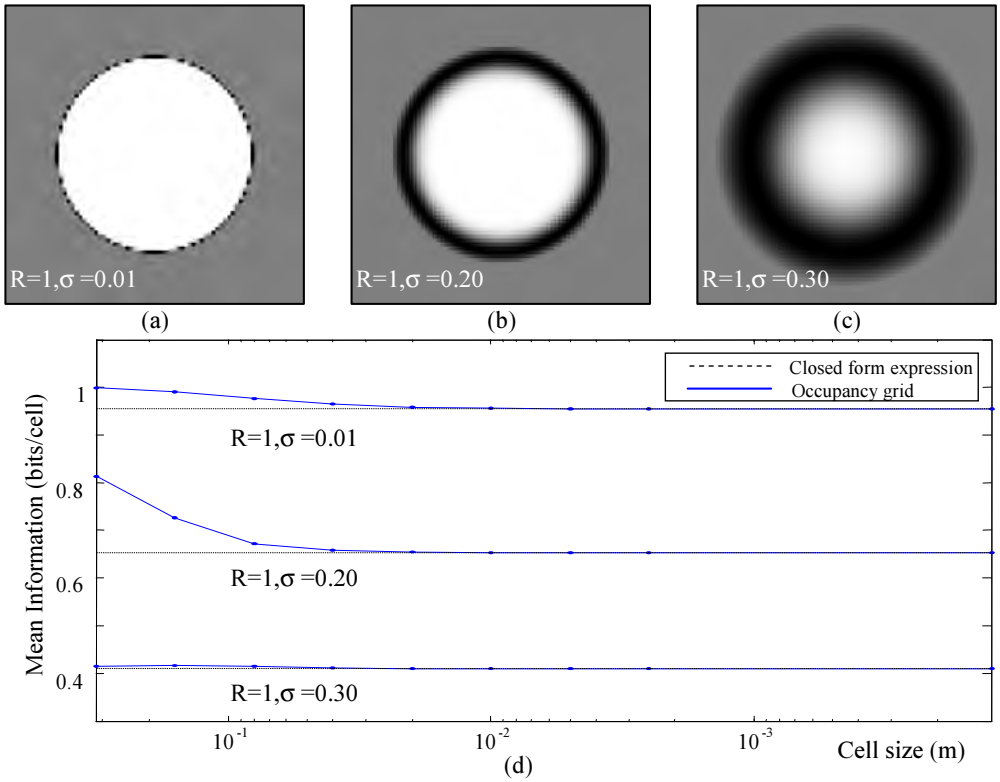


Figure D.1: The MI values for the synthetic scenario discussed in the appendix are shown as continuous plots, together with the theoretical limit (derived in this appendix) as a dashed line. It can be seen how the MI tends toward the computed limit as the resolution gets more fine-grained.

classical $p(m_{xy})$, where ρ is the distance from the origin (where the robot is initially) to a given point x, y. Thus, the map content is same in all directions for the whole range of orientations $]-\pi, \pi]$ from the origin. Initially, there is no prior information about the map, thus $p(m_\rho) = 0.5$ for all the values of ρ. If we consider a range sensor whose measurements are corrupted with additive Gaussian noise with standard deviation σ, its inverse sensor model can be defined as:

$$p\left(m_\rho | \, z\right) = \begin{cases} e^{-\frac{(\rho - z)^2}{2\sigma^2}} & \rho \leq z + \sigma\sqrt{\ln 4} \\ 0.5 & \text{otherwise} \end{cases} \tag{D.0.1}$$

where z represents the sensed range. Such a sensor model was plotted in Figure 10.3.

For the case of just one observation the Bayesian estimation of the map becomes simply $p(m_\rho|z_1) = p(m_\rho|z)$ where z is the actual measurement. If we make the cell size to tend toward zero, the expression for the MI, as defined in Eq. (10.4.4), becomes an integral over the map (which is no more a discrete grid but a continuous surface):

$$\bar{I}\left(m\right) = \frac{\int\int \left(1 - H(m_\rho)\right) d\rho d\theta}{\int_\theta \int_\rho Obs\left(m_\rho\right) \ \rho \ d\rho \ d\theta} \tag{D.0.2}$$

where the binary function $Obs(m_\rho)$ takes the value of 1 for those positions which have been observed, and 0 otherwise. The circular symmetry allows us to discard the integration over since it leads to constant factors in both parts of the quotient. Taking this into account, replacing the integration limits, and solving the denominator, we end up with:

$$\bar{I}\left(m\right) = \frac{\int_0^{z+\sigma\sqrt{\ln 4}} \left[1 - H\left(\exp\left(-\frac{(\rho - z)^2}{2\sigma^2}\right)\right)\right] \ \rho \ d\rho}{\frac{1}{2}\left(z + \sigma\sqrt{\ln 4}\right)^2} \tag{D.0.3}$$

The integral in the numerator has no analytical primitive, but precise values can yet be obtained by numerical integration. To demonstrate the convergence of the grid-based MI measurement towards the continuous solution Eq. (D.0.3) we have generated grids for different R, σ, and resolution values. Some of the obtained grids are shown in Figure D.1(a)–(c), whereas the computed MI values are plotted in Figure D.1(d) together with the theoretical predictions from Eq. (D.0.3). It is clear from these graphs that MI effectively converges as the resolution increases and that this limit can be accurately predicted.

APPENDIX E

CONSISTENCY TEST FOR CANDIDATE PAIRINGS

We are interested in testing the hypothesis that a given pair of features in maps A and B do actually match. This statistical test will rely solely on the rigid-body constraint that dictates that both inter-feature distances d_A^2 and d_B^2, measured in each map, must be equal. Note the usage of squared distances due to convenience during the derivation.

We assume that the uncertainty in the feature points is modeled by a 2-d isotropic Gaussian with a standard deviation of σ, and furthermore the Gaussians for different features are uncorrelated (i.e. the errors are independent). Each of the squared distances d_i^2 is then given by:

$$d_i^2 = |p_{i,1} - p_{i,2}|^2 = (x_{i,1} - x_{i,2})^2 + (y_{i,1} - y_{i,2})^2 \tag{E.0.1}$$

By means of linear uncertainty propagation we can model each d_i^2 as a Gaussian

with mean $\bar{d_i^2}$ and variance $\sigma^2_{d_i^2} = \mathbf{J}\mathbf{\Sigma}\mathbf{J}^\top$, where $\mathbf{J}$ stands for the Jacobian of Eq. (E.0.1) and $\mathbf{\Sigma}$ is the covariance of the feature point coordinates. It is clear that, under our assumption of independence for the error in the coordinates, this covariance amounts to $\sigma^2\mathbf{I}_4$, thus by replacing the values of the Jacobians we obtain:

$$\sigma^2_{d_i^2} = \sigma^2\mathbf{J}\mathbf{J}^\top = 8\sigma^2\bar{d_i^2} \tag{E.0.2}$$

Once defined the distribution of each variable d_i^2, we can finally define the auxiliary variable z as the difference between the two squared distances, that is, $z = d_A^2 - d_B^2$. Under the hypothesis of the pairing to be valid, both distances d_A and d_B should be equal, thus z should be null. This allows us to test the hypothesis with a confidence c by means of the following chi-square test:

$$\chi^2 = \frac{(\bar{d_A^2} - \bar{d_B^2})^2}{8\sigma^2(\bar{d_A^2} + \bar{d_B^2})} < \chi^2_{1,c} \tag{E.0.3}$$

where $\chi^2_{d,c}$ is the inverse of the chi-square cumulative distribution function with d degrees of freedom. In the denominator it has been also used the fact that the variance of z is the sum of the variances of the individual squared distances.

APPENDIX F

THE MOBILE ROBOT PROGRAMMING TOOLKIT (MRPT) PROJECT

Implementing mobile robotics software is a complex task where several heterogeneous factors should be accounted for. On the one hand, the core of most robotics algorithms comprises of pure mathematical operations such as linear algebra, vector and matrix manipulation, or other calculations (e.g. particle filter resampling). This part of the software can be, and usually is, implemented using MATLAB in an offline fashion. On the other hand, access to the robotic hardware or execution time optimizations demand a more efficient and versatile language (e.g. C/C++, Python, etc.).

In an efford to unify the implementation of software for mobile robotics, our group MAPIR [1] started the Mobile Robot Programming Toolkit (MRPT) project. MRPT comprises a set of ready-to-use applications for gathering, manipulating and visualizing datasets, as well as for offline processing of these datasets applying different localization and SLAM methods. Moreover, it is also a collection of libraries (refer to Figure F.1) representing one coherent programming framework where all the algorithms described

[1] http://mapir.isa.uma.es/

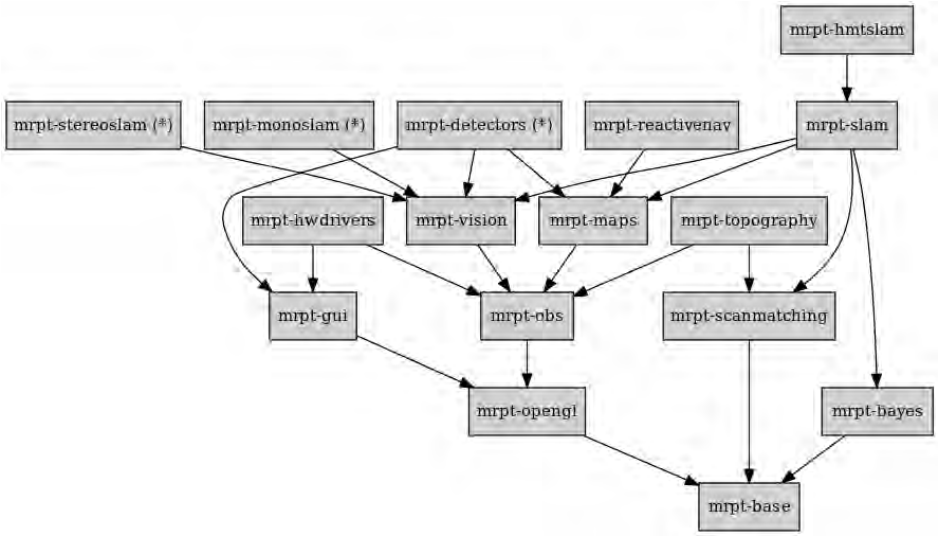


Figure F.1: The structure of the elements within MRPT as a graph of dependencies. Applications depend on one or more MRPT libraries. Those marked with a (*) are experimental or under development libraries.

in this thesis have been implemented. It must be also remarked that MRPT is employed within the BABEL framework [Fer99,FM08] for developing software modules for online robot operation. Work is also in progress to provide MRPT packages for Willow Garage's Robot Operative System (ROS).

The whole toolkit has been developed in C++, with many time critical functions being implemented in assembler (sometimes exploiting the SSE2 capabilities of the processor). The manipulation of images is done through the highly-optimized library OpenCV [2], in whose development the author has also participated in order to achieve a seamless integration with MRPT.

MRPT is free software, released under the GNU General Public License (GPL) version 3. In addition to the official website [3], it can also be found in the official repositories of the most popular GNU/Linux distributions [4].

[2] http://sourceforge.net/projects/opencvlibrary/
[3] http://www.mrpt.org/
[4] At the time of writing this, MRPT is available for Debian, Ubuntu and Fedora.

INDEX

BIBLIOGRAPHY

[Abr65] M. Abramowitz and I. Stegun. *Handbook of mathematical functions*. New York, 1965.

[Als72] D. Alspach and H. Sorenson. *Nonlinear Bayesian estimation using Gaussian sum approximations*. IEEE Transactions on Automatic Control, **volume 17 (4)**, pp. 439–448, 1972.

[Apo07] T. Apostol. *Calculus, Vol. 2: Multi-Variable Calculus and Linear Algebra with Applications*. Wiley India Pvt. Ltd., 2007.

[Ark98] R. Arkin. *Behavior-Based Robotics*. MIT Press, 1998.

[Arr02a] K. Arras, J. Castellanos and R. Siegwart. *Feature-based multi-hypothesis localization and tracking for mobile robots using geometric constraints*. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 2. 2002.

[Arr02b] K. Arras, J. Persson, N. Tomatis and R. Siegwart. *Real-time obstacle avoidance for polygonal robots with a reduceddynamic window*. In *IEEE International Conference on Robotics and Automation*, volume 3. 2002.

[Aru02] M. Arulampalam, S. Maskell, N. Gordon, T. Clapp, D. Sci, T. Organ and S. Adelaide. *A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking*. IEEE Transactions on Signal Processing, **volume 50 (2)**, pp. 174–188, 2002.

[Aya89] N. Ayache and O. Faugeras. *Maintaining representations of the environment of a mobile robot*. IEEE Transactions on Robotics and Automation, **volume 5 (6)**, pp. 804–819, 1989.

[Bai06a] T. Bailey and H. Durrant-Whyte. *Simultaneous localisation and mapping (SLAM): Part II-State of the art*. Robotics and Automation Magazine, **volume 13**, pp. 108–117, 2006.

[Bai06b] T. Bailey, J. Nieto, J. Guivant, M. Stevens and E. Nebot. *Consistency of the EKF-SLAM Algorithm*. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3562–3568. 2006.

[Bal93] T. Balch. *Avoiding the past: a simple but effective strategy for reactivenavigation*. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 678–685. 1993.

[Bay06] H. Bay, T. Tuytelaars and L. Van Gool. *Surf: Speeded up robust features*. Lecture Notes in Computer Science, **volume 3951**, p. 404, 2006.

[Bee05] P. Beeson, N. Jong and B. Kuipers. *Towards Autonomous Topological Place Detection Using the Extended Voronoi Graph*. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 4373–4379. 2005.

[Bel93] B. Bell and F. Cathey. *The iterated Kalman filter update as a Gauss-Newton method*. IEEE Transactions on Automatic Control, **volume 38 (2)**, pp. 294–297, 1993.

[Ben92] J. Benediktsson and P. Swain. *Consensus theoretic classification methods*. IEEE Transactions on Systems, Man, and Cybernetics, **volume 22 (4)**, pp. 688–704, 1992.

[Bes92] P. Besl and N. McKay. *A method for registration of 3-D shapes*. IEEE Transactions on Pattern Analysis and Machine Intelligence, **volume 14 (2)**, pp. 239–256, 1992.

[Bir06] A. Birk and S. Carpin. *Merging occupancy grid maps from multiple robots*. IEEE Proceedings, **volume 94 (7)**, p. 1384, 2006.

[Bis06] C. Bishop *et al. Pattern recognition and machine learning*. Springer New York:, 2006.

[Bla06a] J.-L. Blanco, J.-A. Fernández-Madrigal and J. Gonzalez. *An Entropy-Based Measurement of Certainty in Rao-Blackwellized Particle Filter Mapping*. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3550–3555. 2006.

[Bla06b] J.-L. Blanco, J. Gonzalez and J.-A. Fernández-Madrigal. *Consistent Observation Grouping for Generating Metric-Topological Maps that Improves Robot Localization*. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 818–823. 2006.

[Bla06c] J.-L. Blanco, J. Gonzalez and J.-A. Fernández-Madrigal. *The Trajectory Parameter Space (TP-Space): A New Space Representation for Non-Holonomic Mobile Robot Reactive Navigation*. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1195–1200. 2006.

[Bla07a] J.-L. Blanco. *Online dataset repository*, 2007. URL "http://babel.isa.uma.es/mrpt/downloads/".

[Bla07b] J.-L. Blanco, J.-A. Fernández-Madrigal and J. Gonzalez. *A New Approach for Large-Scale Localization and Mapping: Hybrid Metric-Topological SLAM*. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 2061–2067. 2007.

[Bla07c] J.-L. Blanco, J. Gonzalez and J.-A. Fernández-Madrigal. *A Consensus-based Approach for Estimating the Observation Likelihood of Accurate Range Sensors*. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 4032–4037. 2007.

[Bla07d] J.-L. Blanco, J. Gonzalez and J.-A. Fernández-Madrigal. *Mobile robot egomotion estimation by proprioceptive sensor fusion*. In *International Symposium on Signal Processing and Its Applications*, pp. 1–4. 2007.

[Bla07e] J.-L. Blanco, J. Gonzalez and J.-A. Fernández-Madrigal. *A new method for robust and efficient occupancy grid-map matching*. In *Pattern Recognition and Image Analysis*, volume 4478 of *LNCS*, pp. 194–201. Springer, 2007. ISBN 978-3-540-72848-1.

[Bla08a] J. Blanco, J. Fernández-Madrigal and J. Gonzalez. *Efficient Probabilistic Range-Only SLAM*. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1017–1022. 2008.

[Bla08b] J.-L. Blanco. *Derivation and Implementation of a Full 6D EKF-based Solution to Bearing-Range SLAM*. Technical report, 2008.

[Bla08c] 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, **volume 27 (1)**, pp. 73–89, 2008.

[Bla08d] J.-L. Blanco, J.-A. Fernández-Madrigal and J. Gonzalez. *Towards a Unified Bayesian Approach to Hybrid Metric-Topological SLAM*. IEEE Transactions on Robotics, **volume 24 (2)**, pp. 259–270, 2008.

[Bla08e] J.-L. Blanco, J. Gonzalez and J.-A. Fernández-Madrigal. *A pure probabilistic approach to range-only SLAM*. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 1436–1441. 2008.

[Bla08f] J.-L. Blanco, J. Gonzalez and J.-A. Fernández-Madrigal. *Extending Obstacle Avoidance Methods through Multiple Parameter-Space Transformations*. Autonomous Robots, **volume 24 (1)**, pp. 29–48, January 2008.

[Bla08g] J.-L. Blanco, J. González and J.-A. Fernández-Madrigal. *Foundations of Parameterized Trajectories-based Space Transformations for Obstacle Avoidance*, chapter 2. Mobile Robots Motion Planning: New Challenges. I-Tech Education & Publishing, 2008. ISBN 978-3-902613-35-6. URL `http://babel.isa.uma.es/mrpt/papers/tpspace/`.

[Bla08h] J.-L. Blanco, J. Gonzalez and J.-A. Fernández-Madrigal. *An optimal filtering algorithm for non-parametric observation models in robot localization*. In *IEEE International Conference on Robotics and Automation (ICRA'08)*, pp. 461–466. May 2008.

[Bla09a] J.-L. Blanco. *Resampling methods for particle filtering (MATLAB Central package repository)*, 2009. URL `http://www.mathworks.com/matlabcentral/fileexchange/24968`.

[Bla09b] J.-L. Blanco, J. Gonzalez and J.-A. Fernández-Madrigal. *Subjective local maps for hybrid metric-topological SLAM*. Robotics and Autonomous Systems, **volume 57 (1)**, pp. 64–74, 2009.

[Bla10a] J.-L. Blanco. *The Mobile Robot Programming Toolkit (MRPT) website*, 2010. URL `"http://www.mrpt.org/"`.

[Bla10b] J.-L. Blanco, J. Gonzalez and J.-A. Fernández-Madrigal. *A Robust and Multi-Hypothesis Approach to Occupancy Grid Map Matching*. (Submitted), 2010.

[Bla10c] J.-L. Blanco, J. Gonzalez and J.-A. Fernández-Madrigal. *An Experimental Comparison of Image Feature Detectors and Descriptors applied to Grid Map Matching*. (Submitted), 2010.

[Bon01] D. Bonnafous, S. Lacroix and T. Simeon. *Motion generation for a rover on rough terrains*. In *International Conference on Intelligent Robots and Systems*. 2001.

[Bor89] J. Borenstein and Y. Koren. *Real-time obstacle avoidance for fact mobile robots*. IEEE Transactions on Systems, Man and Cybernetics, **volume 19 (5)**, pp. 1179–1187, 1989.

[Bor91] J. Borenstein and Y. Koren. *The vector field histogram-fast obstacle avoidance for mobilerobots*. IEEE Transactions on Robotics and Automation, **volume 7 (3)**, pp. 278–288, 1991.

[Bor96] J. Borenstein, H. Everett and L. Feng. *Where am I? Sensors and Methods for Mobile Robot Positioning.* University of Michigan, 1996.

[Bor99] H. Borotschnig, L. Paletta and A. Pinz. *A Comparison of Probabilistic, Possibilistic and Evidence Theoretic Fusion Schemes for Active Object Recognition.* Computing, **volume 62 (4)**, pp. 293–319, 1999.

[Bos03] M. Bosse, P. Newman, J. Leonard, M. Soika, W. Feiten and S. Teller. *An Atlas framework for scalable mapping.* In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 2, pp. 1899–1906. 2003.

[Bos04] M. Bosse, P. Newman, J. Leonard and S. Teller. *Simultaneous Localization and Map Building in Large-Scale Cyclic Environments Using the Atlas Framework.* The International Journal of Robotics Research, **volume 23 (12)**, pp. 1113–1139, 2004.

[Bou02] F. Bourgault, A. Makarenko, S. Williams, B. Grocholsky and H. Durrant-Whyte. *Information based adaptive robotic exploration.* In *Proceedings of IEEE International Symposium on Computational Intelligence in Robotics and Automation*, volume 1. 2002.

[Bur97] W. Burgard, D. Fox and S. Thrun. *Active mobile robot localization.* In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI).* Morgan Kaufmann, San Mateo, CA, 1997.

[Bur99] W. Burgard, A. Cremers, D. Fox, D. Hähnel, G. Lakemeyer, D. Schulz, W. Steiner and S. Thrun. *Experiences with an interactive museum tour-guide robot.* Artificial Intelligence, **volume 114 (1-2)**, pp. 3–55, 1999.

[Bur00] W. Burgard, M. Moors, D. Fox, R. Simmons and S. Thrun. *Collaborative multi-robot exploration.* In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 1, pp. 476–481. 2000.

[Cas04] J. Castellanos, J. Neira and J. Tardos. *Limits to the consistency of EKF-based SLAM.* 5th IFAC Symposium on Intelligent Autonomous Vehicles, 2004.

[Cho01a] K. Choo and D. Fleet. *People tracking using hybrid Monte Carlo filtering.* In *Proceedings of IEEE International Conference on Computer Vision*, volume 2, pp. 321–328. July 2001.

[Cho01b] H. Choset and K. Nagatani. *Topological simultaneous localization and mapping (SLAM): Toward Exact Localization Without Explicit Localization.* IEEE Transactions on Robotics and Automation, **volume 17 (2)**, pp. 125–137, 2001.

[Cho05] H. Choset, S. Hutchinson, K. Lynch, G. Kantor, W. Burgard, L. Kavraki and S. Thrun. *Principles of robot motion: theory, algorithms, and implementation*. The MIT Press, 2005.

[Civ08] J. Civera, A. Davison and J. Montiel. *Inverse depth parametrization for monocular SLAM*. IEEE Transactions on Robotics, **volume 24 (5)**, pp. 932–945, 2008.

[Cor03] S. Coradeschi and A. Saffiotti. *An introduction to the anchoring problem*. Robotics and Autonomous Systems, **volume 43 (2-3)**, pp. 85–96, 2003.

[Cov91] T. Cover and J. Thomas. *Elements of information theory*. Wiley New York, 1991.

[Cum07] M. Cummins and P. Newman. *Probabilistic Appearance Based Navigation and Loop Closing*. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 2042–2048. 2007.

[Cum08] M. Cummins and P. Newman. *FAB-MAP: Probabilistic localization and mapping in the space of appearance*. The International Journal of Robotics Research, **volume 27 (6)**, p. 647, 2008.

[Dav07] A. Davison, I. Reid, N. Molton and O. Stasse. *MonoSLAM: Real-Time Single Camera SLAM*. IEEE Transactions on Pattern Analysis and Machine Intelligence, **volume 29 (6)**, pp. 1052–1067, 2007.

[Daw79] A. Dawid. *Conditional independence in statistical theory*. Journal of the Royal Statistical Society. Series B (Methodological), pp. 1–31, 1979.

[Del99] F. Dellaert, D. Fox, W. Burgard and S. Thrun. *Monte Carlo localization for mobile robots*. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 2. 1999.

[Dis01] M. Dissanayake, P. Newman, S. Clark, H. Durrant-Whyte and M. Csorba. *A solution to the simultaneous localization and map building (SLAM) problem*. IEEE Transactions on Robotics and Automation, **volume 17 (3)**, pp. 229–241, 2001.

[Dju06] J. Djugash, S. Singh, G. Kantor and W. Zhang. *Range-only SLAM for robots operating cooperatively with sensor networks*. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 2078–2084. 2006.

[Dju08] J. Djugash, S. Singh and B. P. Grocholsky. *Decentralized mapping of robot-aided sensor networks*. In *Proceedings of the IEEE International Conference on Robotics and Automation*. May 2008.

[Dou00a] A. Doucet, N. de Freitas, K. Murphy and S. Russell. *Rao-Blackwellised particle filtering for dynamic Bayesian networks*. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, pp. 176–183. 2000.

[Dou00b] A. Doucet, S. Godsill and C. Andrieu. *On sequential Monte Carlo sampling methods for Bayesian filtering*. Statistics and Computing, **volume 10 (3)**, pp. 197–208, 2000.

[Dou01] A. Doucet, N. de Freitas and N. Gordon. *Sequential Monte Carlo methods in practice*. Springer, 2001.

[Dou05] R. Douc, O. Cappé and E. Moulines. *Comparison of resampling schemes for particle filtering*. In *Proceedings of the 4th International Symposium on Image and Signal Processing and Analysis*, pp. 64–69. 2005.

[Duc01] T. Duckett and U. Nehmzow. *Mobile robot self-localisation using occupancy histograms and a mixture of gaussian location hypotheses*. Robotics and Autonomous Systems, **volume 34 (2–3)**, pp. 119–130, 2001.

[Dud91] G. Dudek, M. Jenkin, E. Milios and D. Wilkes. *Robotic exploration as graph construction*. IEEE Transactions on Robotics and Automation, **volume 7 (6)**, pp. 859–865, 1991.

[DW06] H. Durrant-Whyte and T. Bailey. *Simultaneous localization and mapping: part I*. IEEE Robotics & Automation Magazine, **volume 13 (2)**, pp. 99–110, 2006.

[Ehr04] H. Ehrig, U. Prange and G. Taentzer. *Fundamental Theory for Typed Attributed Graph Transformation*, volume 3256 of *Lecture Notes in Computer Science*. Springer Berlin, 2004. ISBN 978-3-540-23207-0.

[Elf89] A. Elfes. *Using occupancy grids for mobile robot perception and navigation*. Computer, **volume 22 (6)**, pp. 46–57, 1989.

[Est05] C. Estrada, J. Neira and J. Tardos. *Hierarchical SLAM: Real-Time Accurate Mapping of Large Environments*. IEEE Transactions on Robotics, **volume 21 (4)**, pp. 588–596, 2005.

[Est09] A. Esteban, J. G. Jimenez, A. Ramirez, J.-L. Blanco-Claraco, C. G. Andrades, V. A. Espejo and F. Moreno-Dueñas. *Vehículo terrestre para el análisis topográfico de firmes de infraestructuras lineales (Modelo de utilidad, ES 1 069 405 U)*, 2009.

[Eva92] L. Evans and R. Gariepy. *Measure Theory and Fine Properties of Functions*. CRC Press, 1992.

[Eva01] M. Evans, N. Hastings and B. Peacock. *Statistical distributions*. Measurement Science and Technology, **volume 12**, p. 117, 2001.

[Fei94] W. Feiten, R. Bauer and G. Lawitzky. *Robust obstacle avoidance in unknown and cramped environments*. pp. 2412–2417. 1994.

[Fer99] J. Fernandez and J. Gonzalez. *The NEXUS open system for integrating robotic software*. Robotics and Computer-Integrated Manufacturing, **volume 15 (6)**, pp. 431–440, 1999.

[Fis81] M. Fischler and R. Bolles. *Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography*. Communications of the ACM, **volume 24 (6)**, pp. 381–395, 1981.

[FM00] J.-A. Fernández-Madrigal. *Modeling and Generation of Multiple Abstractions for Representing Large-Scale Space. An Application to Mobile Robots*. Ph.D. thesis, University of Malaga, 2000.

[FM02] J.-A. Fernández-Madrigal and J. Gonzalez. *Multihierarchical graph search*. IEEE Transactions on Pattern Analysis and Machine Intelligence, **volume 24 (1)**, pp. 103–113, 2002.

[FM07] J.-A. Fernández-Madrigal, E. Cruz-Martin, J. Gonzalez, C. Galindo and J.-L. Blanco. *Application of UWB and GPS Technologies for Vehicle Localization in Combined Indoor-Outdoor Environments*. International Symposium on Signal Processing and Its Applications (ISSPA), 2007.

[FM08] J.-A. Fernández-Madrigal, C. Galindo, J. González, E. Cruz-Martín and A. Cruz-Martín. *A software engineering approach for the development of heterogeneous robotic applications*. Robotics and Computer Integrated Manufacturing, **volume 24 (1)**, pp. 150–166, 2008.

[Fon03] T. Fong, I. Nourbakhsh and K. Dautenhahn. *A survey of socially interactive robots*. Robotics and autonomous systems, **volume 42 (3-4)**, pp. 143–166, 2003.

[Fox97] D. Fox, W. Burgard and S. Thrun. *The dynamic window approach to collision avoidance*. IEEE Robotics & Automation Magazine, **volume 4 (1)**, pp. 23–33, 1997.

[Fox99a] D. Fox, W. Burgard, F. Dellaert and S. Thrun. *Monte Carlo localization: Efficient position estimation for mobile robots*. Proc. of the National Conference on Artificial Intelligence (AAAI), **volume 113**, p. 114, 1999.

[Fox99b] D. Fox, W. Burgard and S. Thrun. *Markov localization for mobile robots in dynamic environments*. Journal of Artificial Intelligence Research, **volume 11 (3)**, pp. 391–427, 1999.

[Fox03] D. Fox. *Adapting the Sample Size in Particle Filters Through KLD-Sampling.* International Journal of Robotics Research, **volume 22 (12)**, pp. 985–1003, 2003.

[Fre05] U. Frese. *A Proof for the Approximate Sparsity of SLAM Information Matrices.* In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 331–337. 2005.

[Fre06] U. Frese. *A Discussion of Simultaneous Localization and Mapping.* Autonomous Robots, **volume 20 (1)**, pp. 25–42, 2006.

[Gal04] C. Galindo, J. Fernández and J. Gonzalez. *Hierarchical Task Planning through World Abstraction.* IEEE Transactions on Robotics, **volume 20 (4)**, pp. 667–690, 2004.

[Gal05] C. Galindo, A. Saffiotti, S. Coradeschi, P. Buschka, J.-A. Fernández-Madrigal and J. Gonzalez. *Multi-hierarchical semantic maps for mobile robotics.* In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2278–2283. 2005.

[Gal06] C. Galindo. *A Multi-Hierarchical Symbolic Model for Improving Mobile Robot Operation.* Ph.D. thesis, University of Malaga, 2006.

[Gen86] C. Genest and J. Zidek. *Combining Probability Distributions: A Critique and an Annotated Bibliography.* Statistical Science, **volume 1 (1)**, pp. 114–135, 1986.

[GJ93] J. Gonzalez Jimenez. *Estimación de la Posición y Construcción de Mapas para un Robot Móvil equipado con un Escáner Láser Radial.* Ph.D. thesis, University of Malaga, 1993.

[Goe07] C. Goerick, B. Bolder, H. Janßen, M. Gienger, H. Sugiura, M. Dunn, I. Mikhailova, T. Rodemann, H. Wersing and S. Kirstein. *Towards incremental hierarchical behavior generation for humanoids.* In *IEEE/RAS International Conference on Humanoids*, volume 2007. 2007.

[Gol96] G. Golub, L. Van and F. Charles. *Matrix Computations.* Johns Hopkins University Press, 1996.

[Gon94] J. Gonzalez, A. Ollero and A. Reina. *Map Building for a Mobile Robot equipped with a Laser Range Scanner.* In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 1904–1909. 1994.

[Gon06a] J. Gonzalez, C. Galindo, J.-L. Blanco, M. noz A., A. V. and J.-A. Fernández-Madrigal. *The Robotic Wheelchair SENA Project.* DAAAM International Vienna, 2006.

[Gon06b] J. Gonzalez, A. Muñoz, C. Galindo, J.-A. Fernández-Madrigal and J.-L. Blanco. *A Description of the SENA Robotic Wheelchair*. In *Proceedings of the IEEE Mediterranean Electrotechnical Conference*, pp. 437–440. 2006.

[Gon07] J. Gonzalez, J.-L. Blanco, C. Galindo, A. Ortiz-de Galisteo, J.-A. Fernández-Madrigal, F. Moreno and J.-L. Martinez. *Combination of UWB and GPS for indoor-outdoor vehicle localization*. In *IEEE International Symposium on Intelligent Signal Processing*, pp. 885–890. 2007.

[Gon09a] J. González, J.-L. Blanco, C. Galindo, A. Ortiz-de Galisteo, J.-A. Fernández-Madrigal, F. Moreno and J. Mart'inez. *Mobile robot localization based on ultra-wide-band ranging: A particle filter approach*. Robotics and Autonomous Systems, **volume 57**, pp. 496–507, 2009.

[Gon09b] J. González, C. Galindo, J.-L. Blanco, J.-A. Fernández-Madrigal, V. Arévalo and F.-A. Moreno. *Sancho, a fair host robot. a description*. In *IEEE International Conference on Mechatronics (ICM'09)*. April 2009.

[Gor93] N. Gordon, D. Salmond and A. Smith. *Novel approach to nonlinear/non-Gaussian Bayesian state estimation*. IEEE Proceedings on Radar and Signal Processing, **volume 140 (2)**, pp. 107–113, 1993.

[Gri05] G. Grisetti, C. Stachniss and W. Burgard. *Improving Grid-based SLAM with Rao-Blackwellized Particle Filters by Adaptive Proposals and Selective Resampling*. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 2432–2437. 2005.

[Gri07a] G. Grisetti, C. Stachniss and W. Burgard. *Improved Techniques for Grid Mapping With Rao-Blackwellized Particle Filters*. IEEE Transactions on Robotics, **volume 23**, pp. 34–46, Feb 2007.

[Gri07b] G. Grisetti, G. Tipaldi, C. Stachniss, W. Burgard and D. Nardi. *Fast and accurate slam with rao-blackwellized particle filters*. Robotics and Autonomous Systems, **volume 55 (1)**, pp. 30–38, 2007. ISSN 0921-8890.

[Gui01] J. Guivant and E. Nebot. *Optimization of the simultaneous localization and map-buildingalgorithm for real-time implementation*. IEEE Transactions on Robotics and Automation, **volume 17 (3)**, pp. 242–257, 2001.

[Gut99] J. Gutmann and K. Konolige. *Incremental mapping of large cyclic environments*. In *Proceedings of IEEE International Symposium on Computational Intelligence in Robotics and Automation*, pp. 318–325. 1999.

[Had98] H. Haddad, M. Khatib, S. Lacroix and R. Chatila. *Reactive navigation in outdoor environments using potential fields*. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 2. 1998.

[Hah] D. Hahnel, W. Burgard, D. Fox, K. Fishkin and M. Philipose. *Mapping and localization with RFID technology*. In *2004 IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04*, volume 1.

[Hah02] D. Hahnel, D. Schulz and W. Burgard. *Map building with mobile robots in populated environments*. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 496–501. 2002.

[Hah03] D. Hahnel, W. Burgard, D. Fox and S. Thrun. *An efficient FastSLAM algorithm for generating maps of large-scale cyclic environments from raw laser range measurements*. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1. 2003.

[Har88] C. Harris and M. Stephens. *A combined corner and edge detector*. In *Proceedings of Alvey Vision Conference*, volume 15. 1988.

[Har03] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[Hes09] R. Hess. *C implementation of SIFT feature detector*, 2009. URL `http://web.engr.oregonstate.edu/~hess/`.

[How03] A. Howard and N. Roy. *The robotics data set repository (radish)*, 2003. URL `http://radish.sourceforge.net/`.

[IFR08] IFR. *2007 World Robot Market*, 2008. URL `http://www.ifrstat.org/downloads/2008_executive_summary.pdf`.

[Inc99] I. Inc. *Intel Architecture Software Developer's Manual*. Volume 2: Instruction Set Reference, 1999.

[iRo] *iRobot website*.
http://www.irobot.com/.

[Jen96] F. Jensen. *Introduction to Bayesian networks*. Springer-Verlag New York, Inc. Secaucus, NJ, USA, 1996.

[Jim06] J. G. Jimenez, A. Muñoz-Ramirez, J.-A. Fernandez-Madrigal, C. G. Andrades, J.-L. Blanco-Claraco and V. A. Espejo. *Silla de ruedas robotizada con capacidad operativa autonoma (P200602571)*, 2006.

[Jim07] J. G. Jimenez, J.-M. Lopez-Fernandez, C. G. Andrades, V. A. Espejo, J.-L. Blanco-Claraco, J.-A. Fernandez-Madrigal and G. Ambrosio-Cestero. *Dispositivo móvil compacto para la identificación de vehículos y gestión integral in-situ de estacionamientos (P200701191)*, 2007.

[Jul97] S. Julier and J. Uhlmann. *A new extension of the Kalman filter to nonlinear systems*. Int. Symp. Aerospace/Defense Sensing, Simul. and Controls, pp. 182–193, 1997.

[Jul02] S. Julier. *The scaled unscented transformation*. In *Proceedings of the American Control Conference*, volume 6, pp. 4555–4559. 2002.

[Kal60] R. Kalman. *A new approach to linear filtering and prediction problems*. Journal of Basic Engineering, **volume 82 (1)**, pp. 35–45, 1960.

[Kha97] M. Khatib, H. Jaouni, R. Chatila, J. Laumond and T. LAAS-CNRS. *Dynamic path modification for car-like nonholonomic mobile robots*. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 4. 1997.

[Kit98] J. Kittler, M. Hatef, R. Duin and J. Matas. *On combining classifiers*. IEEE Transactions on Pattern Analysis and Machine Intelligence, **volume 20 (3)**, pp. 226–239, 1998.

[Knu81] D. Knuth. *The art of computer programming: Seminumerical algorithms, volume 2*. Reading, MA: Addison-Wesley, 1981.

[Ko03] J. Ko, B. Stewart, D. Fox, K. Konolige and B. Limketkai. *A practical, decision-theoretic approach to multi-robot mapping and exploration*. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 4, pp. 3232–3238. 2003.

[Kou04] K. Kouzoubov and D. Austin. *Hybrid topological/metric approach to SLAM*. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 1. 2004.

[Kuh97] M.-C. Kuhnmunch. *Reality of robotics*. European cleaning Journal, 1997. URL `http://www.nada.kth.se/~hehu/robo/articles/future3/P_RealRobot.html`.

[Kui90] B. Kuipers and Y. Byun. *A Robot Exploration and Mapping Strategy Based on a Semantic Hierarchy of Spatial Representations*. Artificial Intelligence Laboratory, the University of Texas at Austin, 1990.

[Kui04] B. Kuipers, J. Modayil, P. Beeson, M. MacMahon and F. Savelli. *Local metrical and global topological maps in the hybrid spatial semantic hierarchy*. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 5, pp. 4845–4851. 2004.

[Kul51] S. Kullback and R. Leibler. *On Information and Sufficiency*. The Annals of Mathematical Statistics, **volume 22 (1)**, pp. 79–86, 1951.

[Kwo04] N. Kwok and G. Dissanayake. *An efficient multiple hypothesis filter for bearing-only SLAM*. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1. 2004.

[Lam04] F. Lamiraux, D. Bonnafous and O. Lefebvre. *Reactive path deformation for nonholonomic mobile robots*. IEEE Transactions on Robotics, **volume 20 (6)**, pp. 967–977, 2004.

[Lat91] J. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, 1991.

[Lau93] J. Laumond and P. Soueres. *Metric induced by the shortest paths for a car-like mobile robot*. In *IEEE International Conference on Intelligent Robots and Systems*, pp. 1299–1303. 1993.

[LaV01] S. LaValle and J. Kuffner Jr. *Randomized kinodynamic planning*. The International Journal of Robotics Research, **volume 20 (5)**, pp. 378–400, 2001.

[Laz03] S. Lazebnik, C. Schmid and J. Ponce. *A sparse texture representation using affine-invariant regions*. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2. 2003.

[Lil07] A. J. Lilienthal, A. Loutfi, J. L. Blanco, C. Galindo and J. Gonzalez. *A rao-blackwellisation approach to gdm-slam - integrating slam and gas distribution mapping*. In *Proceedings of the European Conference on Mobile Robots (ECMR)*, pp. 126–131. September 19–21 2007.

[Liu96] J. Liu. *Metropolized independent sampling with comparisons to rejection sampling and importance sampling*. Statistics and Computing, **volume 6 (2)**, pp. 113–119, 1996.

[Liu98] J. Liu and R. Chen. *Sequential Monte Carlo Methods for Dynamic Systems*. Journal of the American Statistical Association, **volume 93 (443)**, pp. 1032–1044, 1998.

[Liu03] Y. Liu and S. Thrun. *Results for outdoor-SLAM using sparse extended information filters*. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 1, pp. 1227–1233. 2003.

[Lou07] A. Loutfi, A. Lilienthal, J.-L. Blanco, C. Galindo and J. Gonzalez. *Integrating SLAM into Gas Distribution Mapping*. In *Proceedings of the IEEE International Conference on Robotics and Automation*. 2007.

[Low99] D. Lowe. *Object recognition from local scale-invariant features*. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2. 1999.

[LP83] T. Lozano-Perez. *Spatial Planning: A Configuration Space Approach*. IEEE Transactions on Computers, **volume 32 (2)**, pp. 108–120, 1983.

[LP87] T. Lozano-Perez. *A simple motion-planning algorithm for general robot manipulators*. IEEE Journal of Robotics and Automation, **volume 3 (3)**, pp. 224–238, 1987.

[Lu97a] F. Lu and E. Milios. *Globally Consistent Range Scan Alignment for Environment Mapping*. Autonomous Robots, **volume 4 (4)**, pp. 333–349, 1997.

[Lu97b] F. Lu and E. Milios. *Robot Pose Estimation in Unknown Environments by Matching 2D Range Scans*. Journal of Intelligent and Robotic Systems, **volume 18 (3)**, pp. 249–275, 1997.

[Luc81] B. Lucas and T. Kanade. *An iterative image registration technique with an application to stereo vision*. Proc. DARPA Image Understanding Workshop, **volume 121**, p. 130, 1981.

[Mar06] J. Martínez, J. González, J. Morales, M. A. and G.-C. A. *Genetic and ICP Laser Point Matching for 2D Mobile Robot Motion Estimation*. Journal of Field Robotics, **volume 23**, January 2006.

[Mat98] M. Matsumoto and T. Nishimura. *Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator*. ACM transactions on modeling and computer simulation, **volume 8 (1)**, pp. 3–30, 1998.

[MC07] R. Martinez-Cantin, N. de Freitas and J. Castellanos. *Analysis of Particle Methods for Simultaneous Robot Localization and Mapping and a New Algorithm: Marginal-SLAM*. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 2415–2420. 2007.

[Mik02] K. Mikolajczyk and C. Schmid. *An affine invariant interest point detector*. In *Proceedings of European Conference on Computer Vision*, volume 1, pp. 128–142. Springer, 2002.

[Min02] J. Minguez, L. Montano and J. Santos-Victor. *Reactive navigation for nonholonomic robots using the ego-kinematic space*. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 3. 2002.

[Min04] J. Minguez and L. Montano. *Nearness diagram (ND) navigation: collision avoidance in troublesome scenarios*. IEEE Transactions on Robotics and Automation, **volume 20 (1)**, pp. 45–59, 2004.

[Min06] J. Minguez, L. Montano and J. Santos-Victor. *Abstracting Vehicle Shape and Kinematic Constraints from Obstacle Avoidance Methods*. Autonomous Robots, **volume 20 (1)**, pp. 43–59, 2006.

[Min09] J. Minguez and L. Montano. *Extending Reactive Collision Avoidance Methods to Consider any Vehicle Shape and the Kinematics and Dynamic Constraints*. IEEE Transactions on Robotics, 2009.

[MM07] O. Martínez-Mozos, R. Triebel, P. Jensfelt, A. Rottmann and W. Burgard. *Supervised semantic labeling of places using information extracted from sensor data*. Robotics and Autonomous Systems, **volume 55 (5)**, pp. 391–402, 2007.

[Mod04] J. Modayil, P. Beeson and B. Kuipers. *Using the topological skeleton for scalable global metrical map-building*. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 2, pp. 1530–1536. 2004.

[Moh91] B. Mohar. *The Laplacian spectrum of graphs*. Graph Theory, Combinatorics, and Applications, **volume 2**, pp. 871–898, 1991.

[Mon02a] M. Montemerlo, S. Thrun, D. Koller and B. Wegbreit. *FastSLAM: A factored solution to the simultaneous localization and mapping problem*. Proceedings of the AAAI National Conference on Artificial Intelligence, pp. 593–598, 2002.

[Mon02b] M. Montemerlo, S. Thrun and W. Whittaker. *Conditional particle filters for simultaneous mobile robot localization and people-tracking*. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 1, pp. 695–701. 2002.

[Mon03a] M. Montemerlo. *FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem With Unknown Data Association*. Ph.D. thesis, University of Washington, 2003.

[Mon03b] M. Montemerlo, S. Thrun, D. Koller and B. Wegbreit. *FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges*. In *Proceedings of the International Joint Conference on Artificial Intelligence*. 2003.

[Mon05] L. Montesano, J. Minguez and L. Montano. *Probabilistic scan matching for motion estimation in unstructured environments*. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3499–3504. 2005.

[Mor85] H. Moravec and A. Elfes. *High resolution maps from wide angle sonar*. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 2. 1985.

[Mor88] H. Moravec. *Sensor Fusion in Certainty Grids for Mobile Robots*. AI Magazine, **volume 9 (2)**, pp. 61–74, 1988.

[Mor07] F. Moreno, J.-L. Blanco and J. Gonzalez. *A probabilistic observation model for stereo vision systems: Application to particle filter-based mapping and localization*. In *Pattern Recognition and Image Analysis*, volume 4477 of *LNCS*, pp. 346–353. Springer, 2007. ISBN 978-3-540-72846-7.

[Mur99] K. Murphy. *Bayesian map learning in dynamic environments*. Advances in Neural Information Processing Systems (NIPS), **volume 12**, pp. 1015–1021, 1999.

[Mur00] R. Murphy. *Introduction to AI Robotics*. MIT Press, 2000.

[Nei01] J. Neira and J. Tardós. *Data association in stochastic mapping using the Joint Compatibility test*. IEEE Transactions on Robotics and Automation, **volume 17 (6)**, pp. 890–897, 2001.

[New03] P. Newman and J. Leonard. *Pure range-only sub-sea SLAM*. Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on, **volume 2**, 2003.

[Nie04] J. Nieto, J. Guivant and E. Nebot. *The hybrid metric maps (hymms): a novel map representation for denseslam*. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 391–396. 2004. ISSN 1050-4729. doi:10.1109/ROBOT.2004.1307181.

[Num03] K. Nummiaro, E. Koller-Meier and L. Van Gool. *An adaptive color-based particle filter*. Image and Vision Computing, **volume 21 (1)**, pp. 99–110, 2003.

[Oku04] K. Okuma, A. Taleghani, N. de Freitas, J. Little and D. Lowe. *A boosted particle filter: Multitarget detection and tracking*. European Conference on Computer Vision, **volume 1**, pp. 28–39, 2004.

[Ols04] E. Olson, J. Leonard and S. Teller. *Robust range-only beacon localization*. Autonomous Underwater Vehicles, 2004 IEEE/OES, pp. 66–75, 2004.

[Pal95] P. Pal and A. Kar. *Mobile robot navigation using a neural net*. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 2. 1995.

[Par62] E. Parzen. *On Estimation of a Probability Density Function and Mode*. The Annals of Mathematical Statistics, **volume 33 (3)**, pp. 1065–1076, 1962.

[Par01] L. Paramonov and H. Lund. *A minimalistic Approach to Humanoids*. In *IEEE/RAS International Conference on Humanoids*. 2001.

[Pas02] M. Paskin. *Thin Junction Tree Filters for Simultaneous Localization and Mapping*. Computer, 2002.

[Paz07] L. Paz, P. Jensfelt, J. Tardós and J. Neira. *EKF SLAM Updates in O(n) with Divide and Conquer SLAM.* In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 1657–1663. 2007.

[Pit99] M. Pitt and N. Shephard. *Filtering Via Simulation: Auxiliary Particle Filters.* Journal of the American Statistical Association, **volume 94 (446)**, pp. 590–591, 1999.

[Pla07] C. Plagemann, K. Kersting, P. Pfaff and W. Burgard. *Gaussian beam processes: A nonparametric bayesian measurement model for range finders.* In *Robotics: Science and Systems (RSS)*. 2007.

[Por06] J. Porta and B. Kröse. *Appearance-based concurrent map building and localization.* Robotics and Autonomous Systems, **volume 54 (2)**, pp. 159–164, 2006.

[Qui93] S. Quinlan and O. Khatib. *Elastic bands: connecting path planning and control.* In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 802–807. 1993.

[Ram01] G. Ramírez and S. Zeghloul. *Collision-free path planning for nonholonomic mobile robots using a new obstacle representation in the velocity space.* Robotica, **volume 19 (05)**, pp. 543–555, 2001.

[Ran06] A. Ranganathan, E. Menegatti and F. Dellaert. *Bayesian Inference in the Space of Topological Maps.* IEEE Transactions on Robotics, **volume 22 (1)**, pp. 92–107, 2006.

[Rem04] E. Remolina and B. Kuipers. *Towards a general theory of topological maps.* Artificial Intelligence, **volume 152 (1)**, pp. 47–104, 2004.

[Ris04] B. Ristic, S. Arulampalam and N. Gordon. *Beyond the Kalman Filter: Particle Filters for Tracking Applications.* Artech House, 2004.

[Roy99] N. Roy, W. Burgard, D. Fox and S. Thrun. *Coastal navigation-mobile robot navigation with uncertainty in dynamic environments.* In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1. 1999.

[RT01] A. Reina Terol. *Navegación de robots móviles mediante escáner láser radial.* Ph.D. thesis, University of Malaga, 2001.

[Rub87] D. Rubin. *A noniterative sampling/importance resampling alternative to the data augmentation algorithm for creating a few imputations when fractions of missing information are modest: The SIR algorithm.* Journal of the American Statistical Association, **volume 82 (398)**, pp. 543–546, 1987.

[Rub88] D. Rubin. *Using the SIR algorithm to simulate posterior distributions.* Bayesian Statistics, **volume 3**, pp. 395–402, 1988.

[Run07] A. Runnalls. *Kullback-Leibler Approach to Gaussian Mixture Reduction.* IEEE Transactions on Aerospace and Electronic Systems, **volume 43 (3)**, pp. 989–999, 2007.

[Rus95a] S. Russell and P. Norvig. *Artificial intelligence: a modern approach.* Prentice-Hall, Inc. Upper Saddle River, NJ, USA, 1995.

[Rus95b] S. Russell and P. Norvig. *Artificial intelligence: a modern approach, chapter 14.* Prentice-Hall, Inc. Upper Saddle River, NJ, USA, 1995.

[Ryb03] P. Rybski, S. Roumeliotis, M. Gini and N. Papanikolopoulos. *Appearance-based minimalistic metric SLAM.* In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, pp. 194–199. 2003.

[Sae05] J. Saez and F. Escolano. *Entropy Minimization SLAM Using Stereo Vision.* In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 36–43. 2005.

[Sae06] P. Saeedi, P. Lawrence and D. Lowe. *Vision-Based 3-D Trajectory Tracking for Unknown Environments.* IEEE Transactions on Robotics, **volume 22 (1)**, pp. 119–136, 2006.

[Saf97] E. Saff and A. Kuijlaars. *Distributing many points on a sphere.* The Mathematical Intelligencer, **volume 19 (1)**, pp. 5–11, 1997. ISSN 0343-6993.

[Sav04] F. Savelli and B. Kuipers. *Loop-Closing and Planarity in Topological Map-Building.* In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 2, pp. 1511–1517. 2004.

[Sch98] C. Schlegel. *Fast local obstacle avoidance under kinematic and dynamic-constraints for a mobile robot.* volume 1. 1998.

[Sch01] D. Schulz, W. Burgard, D. Fox and A. Cremers. *Tracking multiple moving targets with a mobile robot using particle filters and statistical data association.* In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 2. 2001.

[Se01] S. Se, D. Lowe and J. Little. *Local and global localization for mobile robots using visual landmarks.* In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1. 2001.

[She99] C. Shekhar, V. Govindu and R. Chellappa. *Multisensor image registration by feature consensus.* Pattern Recognition, **volume 32 (1)**, pp. 39–52, 1999.

[Shi94] J. Shi and C. Tomasi. *Good features to track*. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 593–600. 1994.

[Shi96] K. Shimoga. *Robot grasp synthesis algorithms: A survey*. The International Journal of Robotics Research, **volume 15 (3)**, p. 230, 1996.

[Shi00] J. Shi and J. Malik. *Normalized Cuts and Image Segmentation*. IEEE Transactions on Pattern Analysis and Machine Intelligence, **volume 22 (8)**, pp. 888–905, 2000.

[Sie04] R. Siegwart and I. Nourbakhsh. *Introduction to autonomous mobile robots*. MIT press, 2004.

[Sim96] R. Simmons. *The curvature-velocity method for local obstacle avoidance*. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 4. 1996.

[Sim05] R. Sim and N. Roy. *Global A-Optimal Robot Exploration in SLAM*. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 661–666. 2005.

[Sin02] S. Singh, G. Kantor and D. Strelow. *Recent results in extensions to simultaneous localization and mapping*. International Symposium on Experimental Robotics, 2002.

[Smi88] R. Smith, M. Self and P. Cheeseman. *A stochastic map for uncertain spatial relationships*. The fourth international symposium on Robotics Research, pp. 467–474, 1988.

[Smi90] R. Smith, M. Self and P. Cheeseman. *Estimating uncertain spatial relationships in robotics*. Autonomous Robot Vehicles, **volume 1**, pp. 167–193, 1990.

[Sog01] T. Sogo, H. Ishiguro and T. Ishida. *Acquisition and propagation of spatial constraints based onqualitative information*. IEEE Transactions on Pattern Analysis and Machine Intelligence, **volume 23 (3)**, pp. 268–278, 2001.

[Sou96] P. Soueres and J. Laumond. *Shortest paths synthesis for a car-like robot*. IEEE Transactions on Automatic Control, **volume 41 (5)**, pp. 672–688, 1996.

[Spa81] E. Spanier. *Algebraic Topology*. Springer, 1981.

[Sta04] C. Stachniss, D. Hahnel and W. Burgard. *Exploration with active loop-closing for FastSLAM*. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 2. 2004.

[Sta05a] C. Stachniss, G. Grisetti and W. Burgard. *Information Gain-based Exploration Using Rao-Blackwellized Particle Filters*. In *Proceedings of Robotics: Science and Systems (RSS)*. 2005.

[Sta05b] C. Stachniss, G. Grisetti and W. Burgard. *Recovering Particle Diversity in a Rao-Blackwellized Particle Filter for SLAM After Actively Closing Loops*. In *Proceedings of the IEEE International Conference on Robotics and Automation*. 2005.

[Sta06] C. Stachniss. *Exploration and Mapping with Mobile Robots*. Ph.D. thesis, Universitat Freiburg, 2006.

[Tam06] H. Tamimi, H. Andreasson, A. Treptow, T. Duckett and A. Zell. *Localization of mobile robots with omnidirectional vision using Particle Filter and iterative SIFT*. Robotics and Autonomous Systems, **volume 54**, pp. 758–765, 2006.

[Tar02] J. Tardos, J. Neira, P. Newman and J. Leonard. *Robust Mapping and Localization in Indoor Environments Using Sonar Data*. The International Journal of Robotics Research, **volume 21 (4)**, pp. 311–330, 2002.

[Ten03] D. Tenne and T. Singh. *The higher order unscented filter*. In *Proceedings of the American Control Conference*, volume 3, pp. 2441–2446. 2003.

[Thr96] S. Thrun and A. Bucken. *Integrating grid-based and topological maps for mobile robot navigation*. Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI-96), pp. 944–950, 1996.

[Thr01a] S. Thrun. *A Probabilistic On-Line Mapping Algorithm for Teams of Mobile Robots*. The International Journal of Robotics Research, **volume 20 (5)**, p. 335, 2001.

[Thr01b] S. Thrun, D. Fox, W. Burgard and F. Dellaert. *Robust Monte Carlo localization for mobile robots*. Artificial Intelligence, **volume 128 (1-2)**, pp. 99–141, 2001.

[Thr02] S. Thrun. *Robotic Mapping: A Survey*. School of Computer Science, Carnegie Mellon University, 2002.

[Thr03] S. Thrun. *Learning Occupancy Grid Maps with Forward Sensor Models*. Autonomous Robots, **volume 15 (2)**, pp. 111–127, 2003.

[Thr04] S. Thrun, Y. Liu, D. Koller, A. Ng, Z. Ghahramani and H. Durrant-Whyte. *Simultaneous Localization and Mapping with Sparse Extended Information Filters*. The International Journal of Robotics Research, **volume 23 (7-8)**, p. 693, 2004.

[Thr05] S. Thrun, W. Burgard and D. Fox. *Probabilistic Robotics*. The MIT Press, September 2005. ISBN 0262201623.

[Tom03] N. Tomatis, I. Nourbakhsh and R. Siegwart. *Hybrid simultaneous localization and map building: a natural integration of topological and metric*. Robotics and Autonomous Systems, **volume 44 (1)**, pp. 3–14, 2003.

[Vek00] O. Veksler. *Image segmentation by nested cuts*. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pp. 339–344. 2000.

[Ven99] M. Vendittelli, J. Laumond and C. Nissoux. *Obstacle distance for car-like robots*. IEEE Transactions on Robotics and Automation, **volume 15 (4)**, pp. 678–691, 1999.

[Ver90] T. Verma and J. Pearl. *Causal networks: Semantics and expressiveness*. In R. D. Shachter, T. S. Levitt, L. N. Kanal and J. F. Lemmer, editors, *Uncertainty in Artificial Intelligence 4*, pp. 69–76. North-Holland, Amsterdam, 1990.

[Vla99] N. Vlassis, Y. Motomura and B. Krose. *An information-theoretic localization criterion for robot map building*. In *Proceedings of International Conference on Machine Learning and Applications*, pp. 1–6. 1999.

[Vla02] N. Vlassis, B. Terwijn and B. Krose. *Auxiliary particle filter robot localization from high-dimensional sensor observations*. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 1. 2002.

[Wal07] M. Walter, R. Eustice and J. Leonard. *Exactly Sparse Extended Information Filters for Feature-based SLAM*. International Journal of Robotic Research, **volume 26 (4)**, pp. 335–359, 2007. ISSN 0278-3649. doi:http://dx.doi.org/10.1177/0278364906075026.

[Wan00] E. Wan and R. Van Der Merwe. *The unscented Kalman filter for nonlinear estimation*. In *Proceedings of the IEEE Adaptive Systems for Signal Processing, Communications, and Control Symposium*, pp. 153–158. 2000.

[Wei73] A. Weir. *Lebesgue integration and measure*. Cambridge University Press, 1973.

[Wil92] H. William, S. Teukolsky, W. Vetterling and B. Flannery. *Numerical Recipes in C: The art of scientific computing*. Cambridge university press New York, NY, USA, 1992.

[Wu05] J. Wu. *Some properties of the normal distribution*. Technical report, Georgia Institute of Technology, 2005. URL `http://www.cc.gatech.edu/~wujx/paper/Gaussian.pdf`.

[Xu02] H. Xu and S. Yang. *Real-time collision-free motion planning of nonholonomic robots using a neural dynamics based approach.* In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 3. 2002.

[Yam98] B. Yamauchi. *Frontier-based exploration using multiple robots.* In *Proceedings of the second international conference on Autonomous agents*, pp. 47–53. ACM Press New York, NY, USA, 1998.

[Yuh00] J. Yuh. *Design and Control of Autonomous Underwater Robots: A Survey.* Autonomous Robots, **volume 8**, pp. 7–24, 2000.

[Zha03] W. Zhao, R. Chellappa, P. Phillips and A. Rosenfeld. *Face recognition: A literature survey.* Acm Computing Surveys (CSUR), **volume 35 (4)**, pp. 399–458, 2003.

[Zha06] L. Zhang, Y. Kim, G. Varadhan and D. Manocha. *Fast c-obstacle query computation for motion planning.* In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pp. 3035–3040. 2006.

[Zit03] B. Zitova and J. Flusser. *Image registration methods: a survey.* Image and Vision Computing, **volume 21 (11)**, pp. 977–1000, 2003.

[Ziv05] Z. Zivkovic, B. Bakker and B. Krose. *Hierarchical map building using visual landmarks and geometric constraints.* In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2480–2485. 2005.

[Ziv06] Z. Zivkovic, B. Bakker and B. Krose. *Hierarchical map building and planning based on graph partitioning.* In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 803–809. 2006.