Optimal Filtering for Non-parametric Observation Models: Applications to Localization and SLAM

Jose-Luis Blanco, Javier González and Juan-Antonio Fernández-Madrigal

The International Journal of Robotics Research 2010 29: 1726 originally published online 12 May 2010
DOI: 10.1177/0278364910364165

The International Journal of
Robotics Research
29(14) 1726–1742
© The Author(s) 2010
Reprints and permission:
sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/0278364910364165
ijr.sagepub.com

Optimal Filtering for Non-parametric Observation Models: Applications to Localization and SLAM

Jose-Luis Blanco, Javier González and Juan-Antonio Fernández-Madrigal

Abstract
In this work we address the problem of optimal Bayesian filtering for dynamic systems with observation models that cannot be approximated properly as any parameterized distribution. In the context of mobile robots this problem arises in localization and simultaneous localization and mapping (SLAM) with occupancy grid maps. The lack of a parameterized observation model for these maps forces a sample-based representation, commonly through Monte Carlo methods for sequential filtering, also called particle filters. Our work is grounded on the demonstrated existence of an optimal proposal distribution for particle filters. However, this optimal distribution is not directly applicable to systems with non-parametric models. By integrating ideas from previous works on adaptive sample size, auxiliary particle filters, and rejection sampling, we derive a new particle filter algorithm that enables the usage of the optimal proposal to estimate the true posterior density of a non-parametric dynamic system. This new filter is better suited, both theoretically and in practice, than previous approximate methods for indoor and outdoor localization and SLAM, as confirmed by experiments with real robots.

Keywords
Estimation theory, SLAM, localization, particle filter, optimal estimation

1. Introduction

Sequential estimation of the state of dynamic, partially observable systems is a problem with innumerable applications to 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 (Liu 1998 and Chen 1998; Doucet et al. 2001; Ristic et al. 2004).

In the scope of robotics there are two prominent applications of Bayesian sequential estimation that have received huge amounts of attention from the research community in the last decade, namely localization and simultaneous localization and mapping (SLAM) (Fox et al. 1999b; Gutmann and Konolige 1999; Dissanayake et al. 2001; Thrun et al. 2001; Thrun 2002; Hahnel et al. 2003; Estrada et al. 2005; Thrun et al. 2005; Grisetti et al. 2007b). The former consists of estimating the pose of a mobile robot within a known environment, whereas in SLAM the map is also estimated while performing self-localization.

In both problems the choice for the representation of the environment determines the probabilistic estimation method that 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 (Dissanayake et al. 2001; Davison et al. 2007). Thus, SLAM with landmark maps can be solved through Gaussian filters such as the extended Kalman filter (EKF) (Julier and Uhlmann 1997) or the unscented Kalman filter (UKF) (Wan and Van Der Merwe 2000). However, there are other types of map representations, such as occupancy grid maps (Moravec and Elfes 1985; Thrun 2003), where these methods are not applicable, forcing a sample-based representation of probability densities. In this case, sequential estimation can be carried out via Monte Carlo simulations, and the associated filtering algorithms receive the generic name of *particle filters* (Doucet et al. 2001). In this work we focus on these occupancy grids, although the proposed

Department of System Engineering and Automation, University of Malaga, Spain

Corresponding author:
Jose-Luis Blanco, Department of System Engineering and Automation, University of Malaga, Malaga 2907, Spain.
Email: jlblanco@ctima.uma.es

method can be also applied to other maps compatible with a sample-based representation of distributions (e.g. gas concentration maps (Loutfi et al. 2007) or topological maps (Ranganathan et al. 2006)). 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 in landmark maps (Neira and Tardós 2001). Their main drawback is that the probabilistic observation model for grid maps can be evaluated only pointwise in a non-parametric form (Thrun 2001; Thrun et al. 2005), in contrast to the analytical models available for landmark maps (Dissanayake et al. 2001; Davison et al. 2007).

Provided that we are able to draw samples according to the system transition model (the robot motion model in our study case) and to pointwise evaluate the observation model, we can sequentially solve localization and SLAM through one of the most basic particle filter (PF) algorithms, the sequential importance sampling (SIS) filter (Rubin 1987), subsequently modified to account for the particle depletion problem (Arulampalam et al. 2002) by means of a resampling step, leading to the SIS with resampling (SIR) filter (Rubin 1988; Gordon et al. 1993). However, the efficiency of these algorithms is greatly compromised by peaky sensor models and outliers, which cause most of the particles to be discarded in a resampling step and lead to *particle impoverishment* or even to the divergence of the filter. For mobile robots this issue typically arises in robots equipped with low-noise sensors such as laser range finders.

A theoretical solution that enables the efficient representation of probability densities through perfectly distributed particles was proposed by Doucet et al. (2000b), 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 (i.e. from which random samples could be drawn), whereas for grid maps we can evaluate it only pointwise (Thrun et al. 2005).

The contribution of this work is a new PF 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. Our method is grounded on previous works related to optimal sampling (Doucet et al. 2000b, a), auxiliary particle filters (APFs) (Pitt and Shephard 1999), rejection sampling (Liu and Chen 1998), and adaptive sample size for robot localization (Fox2003). All of these ideas are discussed throughout the text. Note also that a preliminary version of this work appeared in Blanco et al. (2008).

In the context of mobile robots, the present work 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 in previous works (Montemerlo 2003; Grisetti et al. 2007b).
- To the best of the authors' knowledge, this is the first time in SLAM that the number of particles has been adapted automatically to the uncertainty that is present at each time step. 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).
- Our method is based on the formulation of a general PF, and does not depend on the reliability of scan matching as in previous works. 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 procedure fails, as pointed out by Montemerlo (2003) and Grisetti et al. (2007b).

Like other PF 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, Kalman-like filters (Julier and Uhlmann 1997; Wan and Van Der Merwe 2000) are more efficient.

The rest of the article is outlined as follows. In Section 2 we discuss previous PF algorithms that have been applied to robotics. Our proposal is introduced in Section 3, and a complexity analysis is presented in Section 4. We provide experimental results with real data in Section 6, and finally we highlight some conclusions.

2. Background

In this section we review the underlying ideas of Monte Carlo methods for sequential Bayesian filtering. We focus on the applications of PFs to robot localization and SLAM. For a good introduction to PFs the reader can refer to Arulampalam et al. (2002), while a more exhaustive review of theoretical advances in the field can be found in Doucet et al. (2001) and Ristic et al. (2004).

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 $x^t = \{x_1, \ldots, x_t\}$ denote the sequence of robot poses (the robot *path*) up to time step[1] t. Then, the posterior of the robot pose can be computed sequentially by applying the Bayes rule:

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

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

Under the assumptions of Gaussian distributions and linear systems, the Kalman filter (Kalman 1960) offers a closed-form, optimal solution to Equation (1). Several improvements have been proposed to overcome the assumption of a linear system, leading to the EKF (Julier and Uhlmann 1997) (and its dual the Extended Information Filter (EIF)(Thrun et al. 2004)), the UKF (Julier 2002; Wan and Van Der Merwe 2000), and other higher-order approximations (Tenne and Singh 2003). The EKF has been the predominant approach to localization and SLAM for almost a decade (Dissanayake et al. 2001). However, some drawbacks of this Gaussian filter led to the popularization of PFs for global localization (Fox et al. 1999a), and, more recently, also for mapping (Murphy 1999; Montemerlo et al. 2002a; Grisetti et al. 2007a).

As opposed to parametric probability distributions (e.g. Gaussians and sum of Gaussians), in a PF the estimated distribution of the pose (and the map in SLAM) is represented by a finite set of hypotheses, or *particles*, which are weighted according to *importance sampling*. The simplest PF algorithm is the SIS filter (Rubin 1987), which is described next in the context of robot localization. Concretely, let $\{x^{t,[i]}\}_{i=1}^{M_t}$ denote a set of robot path hypotheses for the time step t, approximately distributed according to the posterior:

$$x^{t,[i]} \sim p(x^t|z^t, u^t), \quad i = 1, \ldots, M_t. \tag{2}$$

Note that virtually all previous PF techniques rely on M_t representing a constant number of particles for all time steps t (we can find an exception in the work by Fox 2003). Since the particles in Equation (2) will not, in general, be distributed exactly according to the true posterior, they are weighted by the so-called *importance weights* $\omega_t^{[i]}$, therefore obtaining an unbiased estimation of the density. The SIS algorithm consists of simulating the Bayes update in Equation (1) by drawing samples for the new robot pose from some *proposal distribution* doucet2000rbp:

$$x_t^{[i]} \sim q(x_t|x^{t-1,[i]}, z^t, u^t) \tag{3}$$

and updating their weights by

$$\omega_t^{[i]} \propto \omega_{t-1}^{[i]} \frac{p(z_t|x_t, x^{t-1,[i]}, z^{t-1}, u^t)p(x_t|x_{t-1}^{[i]}, u_t)}{q(x_t|x^{t-1,[i]}, z^t, u^t)}. \tag{4}$$

The simplest choice for the proposal distribution $q(\cdot)$ is the robot motion model: the *prior* in Equation (1). In this paper we refer to this choice as the *standard proposal*. In this case, widely employed in robotics (Fox et al. 1999a; Montemerlo et al. 2002a; Fox 2003), the weight update in Equation (4) simplifies to the product of the previous weights with the evaluation, at each particle, of the observation model $p(z_t|x^{t,[i]}, z^{t-1}, u^t)$. Note how the SIS filter requires only the ability of drawing samples from the robot motion model and evaluating the observation likelihood pointwise.

In spite of its simplicity, the SIS filter cannot be used in practice. It has been demonstrated that the variance of the weights increases over time (Doucet et al. 2000a), which eventually leads to the degeneracy of the filter. This is the reason for the introduction of the algorithm SIR (Gordon et al. 1993), where a resampling step replaces those particles with low weights by copies of more likely particles.

In the case of map building, Rao–Blackwellized particle filters (RBPFs) are a practical solution for simultaneously estimate both the robot path and the map (Murphy 1999). These RBPFs have been used for landmark maps (FastSLAM (Montemerlo et al. 2002a)), and for occupancy grids (Grisetti et al. 2007a).

However, all of the above PFs suffer from one problem: their efficiency is strongly influenced by the choice of the proposal distribution $q(\cdot)$. The larger mismatch between the proposal and the observation likelihood, the more particles are wasted in non-relevant areas of the state space. In particular, this is the case of mobile robots equipped with accurate sensors such as laser scanners (Grisetti et al. 2007a).

A more efficient approach was presented by Pitt and Shephard (1999) through the APF, which has also been applied to robot localization (Vlassis et al. 2002). In an APF, the process of drawing particles is separated into two steps. First, 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 lookahead resampling is introduced at each step 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 suboptimal solution.

It has been demonstrated by Doucet et al. (2000b) that the variance of the particle weights is minimized by choosing an *optimal proposal distribution*, which incorporates the information of the most recent observation while propagating particles. We must remark that this optimal proposal has been generalized more recently in another work by Doucet et al. (2006) in the form of *block sampling*, where N consecutive observations are jointly used to derive an"N-joint optimal proposal distribution". The application of this technique to SLAM has been explored in Beevers and Huang (2007). While in this paper we focus on the original optimal distribution introduced in Doucet et al. (2000b), our approach and block sampling are complementary, in the sense that both could be employed together. However, this topic requires further research and is not addressed here.

Regarding the optimal proposal distribution, there exists a closed-form solution for landmark maps which has been reported with the name of FastSLAM 2.0 (Montemerlo 2003). However, for other map representations such as occupancy grids, parametric observation models are not available. A solution proposed by Grisetti et al. (2007a, b) overcomes this by approximating the sensor model with

Table 1. Bayesian Filtering Algorithms that have been Applied to Localization and SLAM.

Proposal distribution	System models	Algorithms
–	Linear Gaussian	Kalman Filter, Kalman, 1960
–	Non-linear Gaussian	EKF, Julier and Uhlmann, 1997
		UKF, Wan and Van Der Merwe, 2000
Standard	Non-linear non-Gaussian	SIR, Gordon et al 1993; APF, Pitt and Shephard, 1999
		RBPF, Murphy, 1999; FastSLAM, Montemerlo et al, 2002
Optimal	Non-linear Gaussian	FastSLAM 2.0, Montemerlo et al 2003; Grisetti et al, 2007a, b
Optimal	Non-linear non-Gaussian	This work

a Gaussian whose mean value is obtained by scan matching over the grid map. This approximation has demonstrated its practical utility allowing the efficient mapping of large environments. However, we should highlight some important drawbacks of this approach. First, the observation likelihood may not be appropriately approximated by a Gaussian in many situations, thus the posterior distribution would be severely distorted. Even in those cases where the observation likelihood could resemble a Gaussian, it cannot be proven that the mean value of the posterior could match with 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 corresponding observations (Grisetti et al. 2007a), but observe that even in those cases we could obtain a more precise posterior by integrating all of the available information, which is lost otherwise. Second, the prior distribution is ignored due to its inaccuracy in comparison with the observation likelihood. However, in an 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.

A classification of the methods discussed in this section is presented in Table 1, which also includes our method for comparison. While this article focuses on localization and SLAM, we should remark that the generic PF presented here can be applied to any other estimation problem where non-parametric observation models appear. For example, in the robotics and computer vision literature we can find several applications of PFs for tracking people (Choo and Fleet 2001; Schulz et al. 2001; Montemerlo et al. 2002b) or arbitrary objects on a sequence of images (Nummiaro et al. 2003; Okuma et al. 2004).

3. The Optimal Particle Filter

3.1. Preliminary Definitions

It has been shown that the optimal proposal distribution that minimizes the variance of the next weights for any generic PF is given by (Doucet et al. 2000b)

$$x_t^{[i]} \sim q(x_t|x^{t-1,[i]}, z^t, u^t) = p(x_t|x^{t-1,[i]}, z^t, u^t)$$
$$= \frac{p(z_t|x_t, x^{t-1,[i]}, z^{t-1}, u^t)p(x_t|x^{t-1,[i]}, z^{t-1}, u^t)}{p(z_t|x^{t-1,[i]}, z^{t-1}, u^t)}. \quad (5)$$

For mobile robots this proposal requires drawing samples from the product of the transition (robot motion) and observation models, which are the terms that appear in the numerator of Equation (5). Since the system state for the last time step (x_t) does not appear in the denominator, this is a constant value μ for each particle i. Therefore, drawing samples from the optimal proposal is equivalent to drawing from

$$x_t^{[i]} \sim \frac{1}{\mu}\overbrace{p(z_t|x_t, x^{t-1,[i]}, z^{t-1}, u^t)}^{\text{Observation model}}\overbrace{p(x_t|x^{t-1,[i]}, z^{t-1}, u^t)}^{\text{Transition model}}. \quad (6)$$

By replacing this optimal proposal in the general equation for the weight update in a SIS filter, in Equation (4), we obtain

$$\omega_t^{[i]} \propto \omega_{t-1}^{[i]} p(z_t|x^{t-1,[i]}, z^{t-1}, u^t). \quad (7)$$

At this point, we state that the purpose of our optimal PF algorithm is to generate samples exactly distributed according to the density in Equation (5), while dynamically adapting the number of samples to assure a good representation of the true posterior at each moment.

To avoid the problem of particle depletion we have found two different approaches in other works. The first is to resample particles at every time step as required to ensure that they represent the true posterior well. Another solution consists of resampling only when a measure of the representativeness of the samples is below a given threshold (Rubin 1988). We employ the first approach for the derivation of our optimal algorithm. As discussed later on, this generic optimal filter fits perfectly to the problem of mobile robot localization. In a subsequent section we introduce a variation applicable to SLAM by using selective resampling in order to avoid problems that arise from the higher dimensionality of that problem.

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. To clarify the exposition we have summarized the process graphically in Figure 1.

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

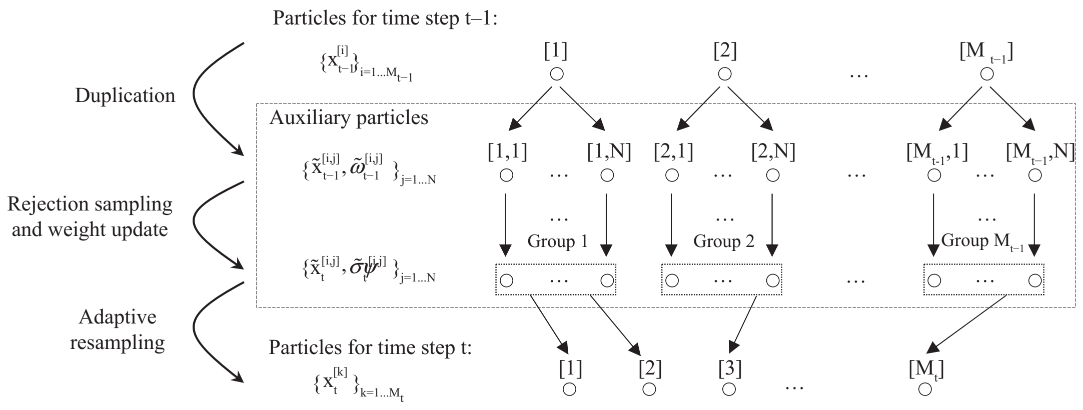


Fig. 1. The theoretic model of our optimal particle filter. An initial set of M_{t-1} particles is first replicated into a set of auxiliary particles, which are then propagated according to the optimal proposal distribution (simulated by rejection sampling). Then, a resampling stage (with an adaptive 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 of the final particles have equal importance weights (omitted in the graph by this reason).

$$x_{t-1}^{[i]} \sim p(x_{t-1}|z^{t-1}, u^{t-1}). \tag{8}$$

Since these samples are optimally distributed, all of them will have equal importance weights, and so they can be omitted. The assumption of perfectly distributed particles for the previous time step is not a problem but for the first iteration of the filter. Typical assumptions for the initial belief 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

$$\begin{aligned} \tilde{x}_{t-1}^{[i,j]} &= x_{t-1}^{[i]}, \quad j = 1, \ldots, N, \\ \tilde{\omega}_{t-1}^{[i,j]} &= \frac{1}{NM_{t-1}}. \end{aligned} \tag{9}$$

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

The auxiliary particles are propagated according to the optimal proposal, in Equation (6), in order to obtain a large (ideally an infinite) number of optimally distributed particles $\tilde{x}_t^{[i,j]}$, from which we finally keep only the required particles for providing a good representation of the posterior. This is achieved by generating the new set of particles $x_t^{[k]}$ by resampling the set of auxiliary samples $\tilde{x}_t^{[i,j]}$.

The key point that allows us to directly generate the optimally distributed particles without computing all of the auxiliary particles is that all of the auxiliary particles $\tilde{x}_t^{[i,j]}$ that come from a given particle $x_{t-1}^{[i]}$ *will have equal 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 Equation (7). These groups of equally weighted samples are schematically represented in Figure 1.

As has been mentioned, to generate the optimal particles $x_t^{[k]}$ we must resample the auxiliary set at time step t. Similarly to the auxiliary PF (Pitt and Shephard 1999), we perform this by drawing indexes i of particles for the previous time step, in our case with a probability proportional to the weights $\tilde{\omega}_t^{[i,j]}$, which are given by

$$\tilde{\omega}_t^{[i,j]} = \tilde{\omega}_{t-1}^{[i,j]} p(z_t|x^{t-1,[i]}, z^{t-1}, u^t). \tag{10}$$

Here the *a priori* likelihood of the observation z_t can be expanded using the law of total probability:

$$\begin{aligned} &p(z_t|x^{t-1,[i]}, z^{t-1}, u^t) \\ &\quad = \int p(x_t|x_{t-1}^{[i]}, u_t) p(z_t|x_t, x^{t-1,[i]}, z^{t-1})\, dx_t. \end{aligned} \tag{11}$$

The terms that appear inside the integral above are the system transition and observation models, respectively. Since we are assuming in this work that we can only draw samples from the system transition model and evaluate pointwise the observation model, a Monte Carlo approximation of the integral $\hat{p}(z_t|\cdot) \approx p(z_t|\cdot)$ can be obtained by means of

$$\hat{p}(z_t|x^{t-1,[i]}, z^{t-1}, u^t) = \frac{1}{B} \sum_{n=1}^{B} p(z_t|x_t^{[n]}, x^{t-1,[i]}, z^{t-1}) \tag{12}$$

with the 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 heuristic parameter of our algorithm, and will be typically in the range 10–200 depending on the specific problem addressed by the filter.

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

```
1: for all particle $x_{t-1}^{[i]}$ do
2:   for $n = 1$ to $B$ do // Generate a set of B samples from the transition model
3:     $x_t^{[n]} \sim p(x_t|x_{t-1}^{[i]}, u_t)$
4:   end for
5:   Use the samples to compute $\hat{p}(z_t|\cdot)$ and $\hat{p}_{max}(z_t|\cdot)$
6:   Compute $\tilde{\omega}_t^{[i]}$ using Equations (10)–(12)
7: end for
8: for $k = 1$ to $M_t\left(\{x_t^{[j]}\}_{j=1}^{k}\right)$ do //$M_t$ dynamically determined by KLD sampling (Fox 2003)
9:   Draw an index $i$ with probability proportional to $\tilde{\omega}_t^{[i]}$.
10:  repeat // Generate a new sample by rejection sampling
11:    $x_t^{[k]} \sim p(x_t|x_{t-1}^{[i]}, u_t)$ // Draw a candidate sample from the transition model
12:    Compute $\Delta$ through Equation (13)
13:    $a \sim Unif(0, 1)$ // Draw a random uniform sample
14:  until $a < \Delta$ // Candidate is accepted with a probability of $\Delta$
15: end for
```

In general, larger values of B would lead to a better approximation of the optimal proposal, but would also carry a higher computational cost. If a grounded value for B is desired for a specific problem, simulations should be performed to determine how fast does the sum in Equation (12) converges.

Going back to the resampling of the auxiliary particles, for each drawn index i we generate a new optimal particle by taking the value of *any* auxiliary particle in the ith group, since all of them have equal probability of being selected in the resampling. That is, the new optimal particle $x_t^{[k]}$ is a copy of $\tilde{x}_t^{[i,j]}$, where the value of j does not need to be specified. Note that the importance weights of the final particles given by our algorithm can be ignored, since particles obtained by resampling all have exactly the same weights.

We need to provide a method 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 Equation (6). 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) (Liu and Chen 1998):

$$\Delta = \frac{p(z_t|x_t^{[k]}, x^{t-1,[i]}, z^{t-1}, u^t)}{\hat{p}_{\max}(z_t|x_t, x^{t-1,[i]}, z^{t-1}, u^t)}. \qquad (13)$$

We must remark that this technique has a stochastic complexity (a random execution time), as discussed in more detail in Section 4. The only quantity required to evaluate Equation (13) is the maximum value of the observation model $\hat{p}_{max}(z_t|\cdot)$. This value can be estimated simultaneously to the Monte Carlo approximation in Equation (12) 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 according to the true posterior given the set of particles for the previous time step. The above 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 method introduced by Fox (2003). In that study the concept of Kullback–Leibler distance (KLD) (Cover and Thomas 1991) was used, which measures the similarity between a pair of probability densities $p_1(x)$ and $p_2(x)$, and is defined as

$$D(p_1, p_2) = \int p_1(x) \log \frac{p_1(x)}{p_2(x)}\, dx. \qquad (14)$$

As that work shows, the minimum number of particles M_t to ensure 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

$$M_t\left(\{x_t^{[j]}\}_j\right) = \frac{1}{2\epsilon}\chi^2_{l-1,1-\delta}, \qquad (15)$$

where $\chi^2_{d,q}$ denotes the qth quantile of the chi-squared distribution with d degrees of freedom. In this approach 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. Refer to Fox (2003) for further details.

In the localization experiments presented later on, we have employed bins of $7cm \times 7cm \times 2°$ in order to discretize the three-dimensional robot pose and obtain the number of occupied bins l for Equation (15), with parameters $\epsilon = 0.01$ and $\delta = 0.01$. In the case of SLAM, note that the whole robot path has an increasing dimensionality of $3t$ for time step t, hence discretizing the robot path into bins

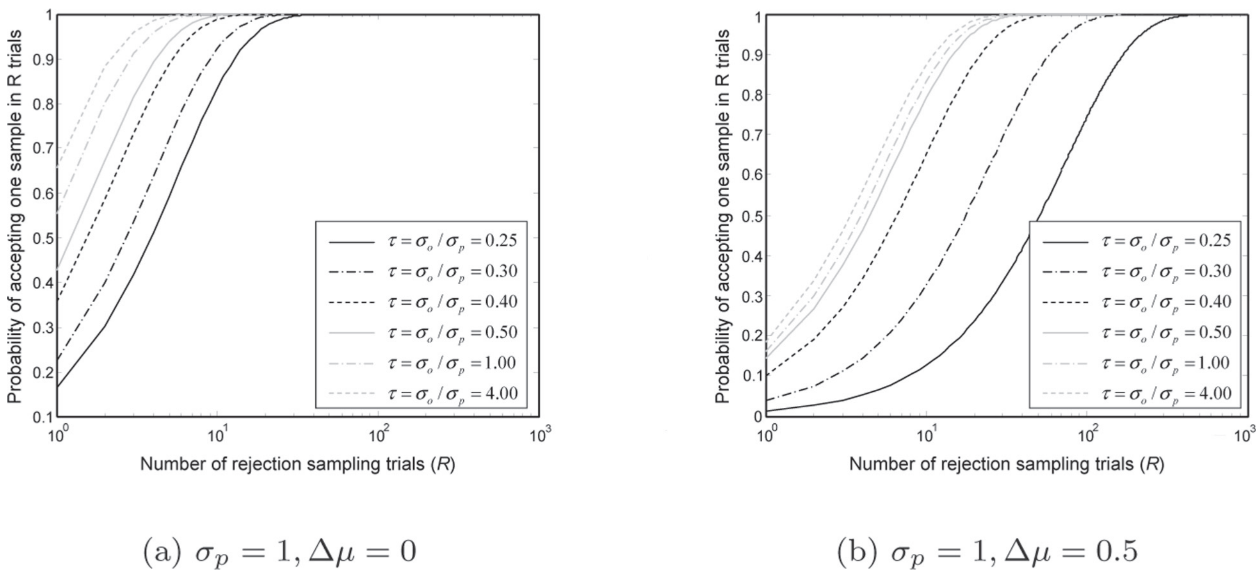


Fig. 2. The probability of the required number of rejection sampling iterations (R) until the first accepted sample, modeled as a Bernoulli process and for a prior and an observation model normally distributed. In (a) the centers of both Gaussians coincide ($\Delta\mu = 0$), whereas in (b) they do not. As a consequence more trials are required in average to obtain an accepted sample. The parameter $\tau = \sigma_o / \sigma_p$ reflects the relative variance of each Gaussian. Observe how more trials are needed as the observation model becomes more peaked (lower τ values).

will probably lead to a rapid exponential growth as the robot explores new areas. We have experimentally verified that considering the latest robot pose only instead of the whole path for computing l is an acceptable approximation while keeping the number of samples reasonably bounded.

To summarize the introduction of our algorithm we present an algorithmic description of the overall method in Algorithm 1.

4. Complexity Analysis

In this section we analyze rigourously the time complexity of our optimal filter, using as a reference the sequence of steps in the algorithmic description of Algorithm 1. Recall 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 Equation (12). We detail next the contributions of the individual operations to the overall execution time:

- *Estimation of $\hat{p}(z_t|\cdot)$ and $\hat{p}_{max}(z_t|\cdot)$* (lines 1–7 in Algorithm 1). Here the observation model is evaluated B times for each particle in $t-1$, thus the complexity becomes $\mathcal{O}(T_L B M_{t-1})$, where T_L denotes the constant time factor associated with a single pointwise evaluation of the observation model.
- *Determination of M_t by means of KLD sampling* (part of line 8). 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 T_K) by implementing the bin counters as a simple array, as suggested in (Fox 2003). Thus, the complexity of the step is $\mathcal{O}(T_K M_t)$. 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 9). This operation requires the computation of the cumulative density function (cdf) of the particle weights, then draw a uniform random number and look up (with $\mathcal{O}(T_C M_{t-1})$) that cdf to obtain an index of particles from the previous set. This process is repeated for each of the M_t particles, thus the overall complexity becomes $\mathcal{O}(T_C M_t M_{t-1})$.
- *Rejection sampling* (lines 10–14). If we use R to denote the number of trials required to obtain an accepted sample in each of the M_t iterations, we have a complexity of $\mathcal{O}(T_L R M_t)$ where T_L is included since the observation model is evaluated at every rejection sampling trial. Since R is actually a random variable this operation has a non-deterministic time complexity (further discussed below).

For the application stressed in this work, a mobile robot with a laser scanner and occupancy grid maps, the overall complexity is strongly determined by the number of times the observation model is evaluated, since T_L will be usually larger than the other time constants. From the individual complexities described above we conclude that this number of times is of the order $\mathcal{O}(BM_{t-1} + RM_t)$. The number 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 loops, decreasing after

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

```
1: for all particles $x_{t-1}^{[i]}$ do
2:   for $n = 1$ to B do // Generate a set of B samples
3:     $x_t^{[n]} \sim p(x_t|x_{t-1}^{[i]}, u_t)$
4:   end for
5:   Use the samples to compute $\hat{p}(z_t|\cdot)$ and $\hat{p}_{max}(z_t|\cdot)$
6: end for
7: $p \leftarrow \text{perm}([1, 2, \ldots, M_{t-1}])$ // Random permutation of the $M_{t-1}$ indices
8: $doResampling \leftarrow ESS(\{\omega^{[i]}\}_{i=1}^{M_{t-1}}) < 0.5$// Selective resampling
9: for $k = 1$ to $M_t\left(\{x_t^{[j]}\}_{j=1}^{k}\right)$ do // $M_t$ dynamically determined by KLD sampling (Fox2003)
10:   if doResampling then
11:     Draw an index $i$ with probability given by $\tilde{\omega}_t^{[i]}$. See Equation (10)
12:   else
13:     if $k \leq M_{t-1}$ then
14:       $i \leftarrow p[k]$// Deterministic sampling
15:     else
16:       $i \sim Unif(1, M_{t-1})$// Uniform (integer) sampling
17:     end if
18:   end if
19:   repeat // Generate a new sample from $i$ by rejection sampling
20:     $x_t^{[k]} \sim p(x_t|x_{t-1}^{[i]}, u_t)$// Draw a candidate sample
21:     $a \sim Unif(0, 1)$ // Draw a random uniform sample
22:   until $a < \Delta$// Candidate accepted with a probability $\Delta$ –see Equation (13)
23:   if doResampling then // Now, assign the weight
24:     $\omega^{[k]} \leftarrow 1$ // Reset weights on resampling
25:   else
26:     $\omega^{[k]} \leftarrow \omega^{[k]} \cdot \tilde{\omega}_t^{[i]}$ // Apply likelihood factor
27:   end if
28: end for
```

closing them. Thus, any bound to the computation time limits the size of the loops our algorithm can process in real-time, just as also happens to EKF-based methods. A promising approach to overcome these limitations is to consider hierarchical (or hybrid) map representations (Bosse et al. 2004; Estrada et al. 2005; Blanco et al. 2007), an issue out of the scope of this article. The other values that determine the performance of our algorithm are B and R. Since B is a fixed parameter, we focus next on the factors that determine the random variable R, i.e. the number of trials required to accept one sample in rejection sampling.

First, 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. The number of Bernoulli trials required to obtain the first success, denoted by R, is known to follow a geometric probability distribution $P(R) = p(1-p)^{R-1}$, where the parameter p states the probability of success for each individual trial. Provided with this parameter, the expected number of trials until the first success is then given by $E[R] = 1/p$ (the mean of a geometric distribution), which determines the average performance of our algorithm.

Unfortunately, the parameter p cannot be computed in closed form for generic prior and observation models. In general, this probability will be high if the 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 the Appendix) for the specific situation of both the prior and the observation model being Gaussians. The cdf of R, given by $cdf(R) = 1 - (1-p)^R$, is represented in Figure 2 for values of p in two different situations: the prior and the observation likelihood being centered at the same point ($\Delta\mu = 0$, left graph), and separate ($\Delta\mu = 0.5$, right graph). It is clear how the first case requires fewer trials than the later for a given cdf of succeeding. For both situations 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 $\Delta\mu = 0.5$.

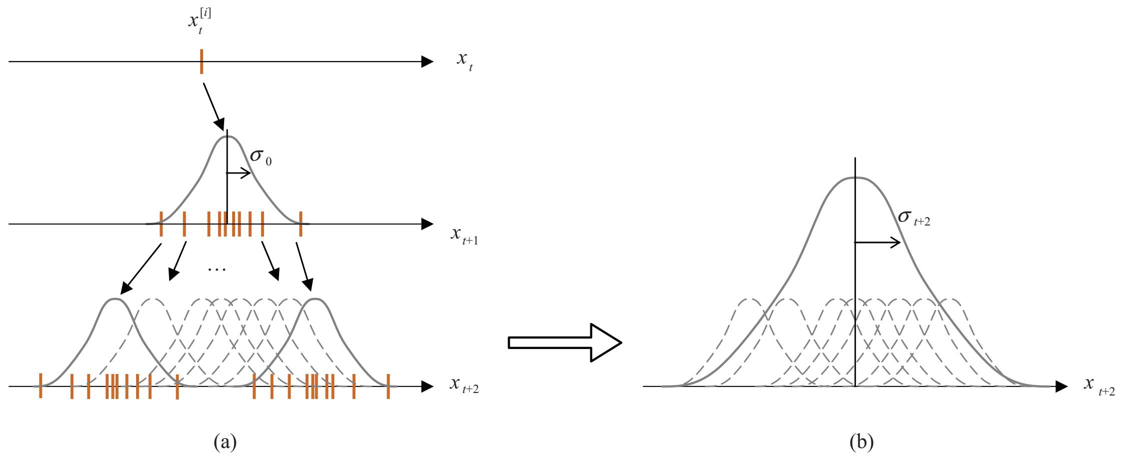


Fig. 3. (a) The number of particles required to maintain a proper sampling of the whole state space in full SLAM grows exponentially with time. (b) An alternative proposed in the text, where only the marginal for the latest robot pose is employed to determine the number of required samples.

As an interesting conclusion, we can state that the time complexity of our algorithm increases with increasing mismatch between the prior and the most recent observation. In other words, the optimal PF 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, 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 losing the optimality in the distribution of samples.

5. Optimal Filtering for RBPF-SLAM

The generic optimal PF algorithm introduced above can be directly applied to robot localization. In this section we propose two modifications to the generic algorithm to improve its suitability to SLAM. First, we recall that the dimensionality of the state space being estimated in the“full SLAM” problem (Thrun et al. 2005) continuously increases over time. In this context, the resampling that our algorithm performs at every step may lead to a loss of the diversity of particles for representing the robot path. This is a common problem of all PFs, and can be alleviated by introducing *selective* resampling steps (Liu and Chen 1998; Grisetti et al. 2007a), as explained below. Second, we also discuss an approximation to the original KLD sampling method (Fox 2003) for determining the dynamic number of samples.

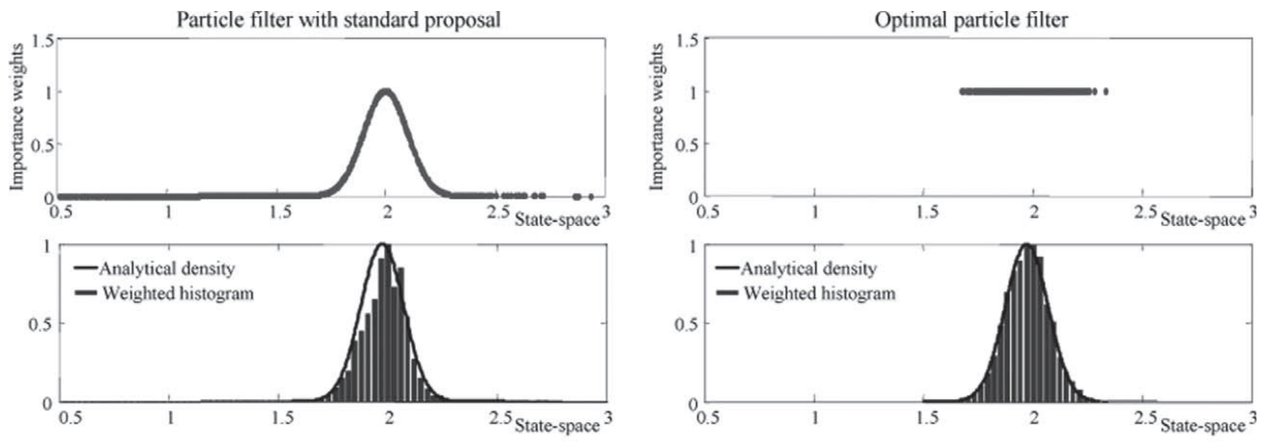
5.1. Selective Resampling

Our purpose is to delay resampling as much as possible to avoid the loss of diversity, for example, by monitoring a measure of diversity such as the effective sample size (ESS) (Liu 1996). At each iteration, we compute the ESS and, only if it is below a threshold (typically 0.5 times the number of particles), the particles are resampled. In that case the procedure will be exactly as described in Algorithm 1. Otherwise particles are not resampled, and their weights must be updated according to the corresponding equation for the optimal proposal in Equation (7), also considering the approximation of $\hat{p}(z_t|\cdot)$ described in Equation (12).

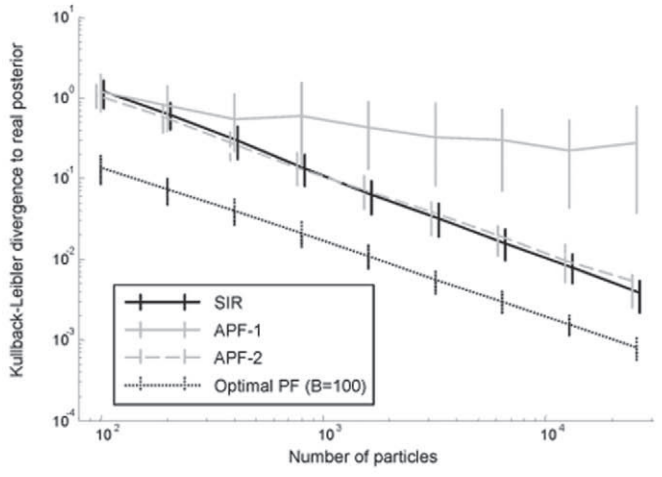
In this latter case, the introduction of a variable number of particles in our algorithm makes the method to draw particle indexes (the i in line 9 of Algorithm 1) more complex, since it must be ensured that all particles have the same probability of passing to the next iteration if resampling is not performed. In a traditional filter with a fixed sample size this is achieved by propagating just once each particle from the previous time step (Liu and Chen 1998; Grisetti 2007a). In our case we propose the following mechanism to generate the particle indexes $i[l]$, for $l = 1, \ldots, M_t$ (recall that indexes i indicate which particles from the previous time step $t-1$ are selected to generate the M_t new particles):

$$i[l] : \begin{cases} i[l] = p[l] & l \leq M_{t-1}, \\ i[l] \sim \text{Uni}(1, M_{t-1}) & l > M_{t-1}, \end{cases} \tag{16}$$

with $p[l] = \text{perm}([1, \ldots, M_{t-1}])$ being a random permutation[2] of the vector $[1, \ldots, M_{t-1}]$, which is computed only once for all of the indexes $i[l]$. In this way, if the number of particles is reduced at some time step, the permutation ensures that all particles have identical probabilities of being removed. If the number of particles remains constant this method only introduces a reordering of the samples, which does not affect the estimated density. Finally, in the case of more particles it is also ensured that all of 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



(a)



(b)

Fig. 4. A comparison of our method to other particle filter (PF) algorithms for a linear, Gaussian system. (a) The obtained particles and weights for two of the algorithms (top), and the weighted histograms of the samples compared with the exact Gaussian density being estimated (bottom). Only the particles for sequential importance sampling with resampling (SIR) and our optimal PF are shown to save space. (b) The average Kullback–Leibler divergence (and 10–90% confidence intervals) between the real and the estimated distributions for the four algorithms. Observe how our method achieves the lowest distance, i.e. the highest similarity, for each sample size.

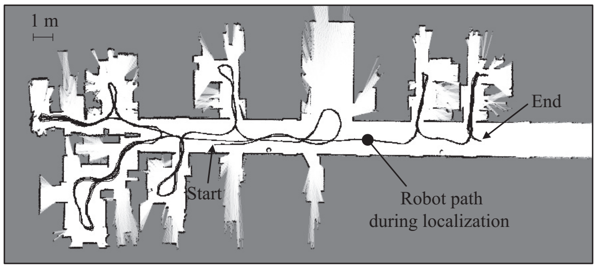
lost as long as $M_t \geq M_{t-1}$ (otherwise it would mean that there are too many particles and we really want to remove some hypotheses). The complete modified algorithm for selective resampling is shown in Algorithm 2.

5.2. Approximate KLD Sampling

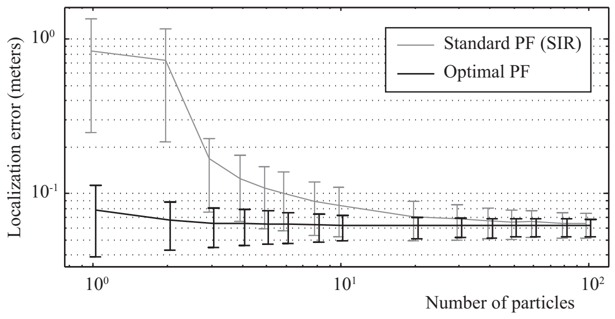
In KLD sampling (Fox 2003), the continuous d-dimensional state space of the PF is approximated by a d-dimensional discrete grid where the number of occupied bins settles the final number of required particles.

In the original context of KLD sampling, robot localization, the dimensionality of the problem is fixed to $d = 3$ (two-dimensional position plus heading). Under such a bounded dimensionality we can expect a limited number of samples in most common situations. A preeminent (although transient) exception is global localization. In contrast, when KLD sampling is applied to full SLAM the problem dimensionality increases with time t since the filter estimates the whole robot *path*, with a dimensionality of dt, that is, of the order of $O(t)$.

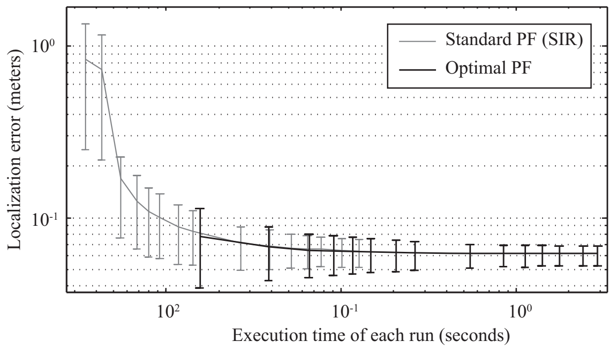
When the number of samples is dynamically adapted in such a problem, it is inevitable for the required number of



(a)



(b)



(c)

Fig. 5. Results for the localization experiments. (a) The map used for the experiment and the robot path through the environment. (b) The average error in positioning while tracking the robot pose using a sequential importance sampling with resampling (SIR) filter and our optimal algorithm, both for different sample sizes. The mean and the 10–90% confidence intervals for 100 repetitions are shown. Observe how our method performs well even for just one particle. (c) The same errors, shown as a function of the overall execution time.

particles to grow without bounds. This is nothing else than the need to perform importance sampling properly on each of the dimensions, which intuitively can be seen to demand an *exponential* number of samples (this is, in fact, mathematically proven in the following). 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 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 PF for SLAM: (i) employ KLD sampling over the whole robot path; or (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 path followed by the robot. First, we assume that the estimates for the d dimensions of the robot pose are uncorrelated between them, thus the problem is equivalent to estimating d unidimensional variables. Focusing on the case of full SLAM, let us consider the evolution along time of one single particle, such as that represented in Figure 3 (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

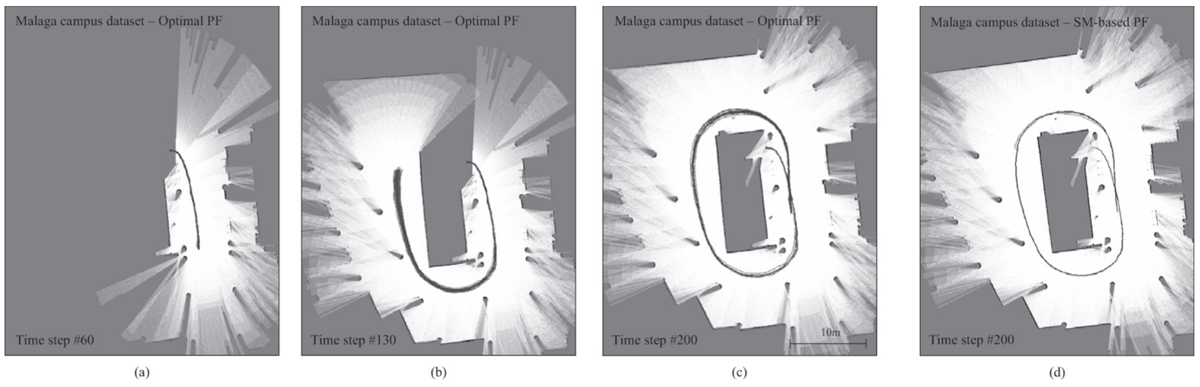

Malaga campus dataset – Optimal PF
Time step #60
(a)
Malaga campus dataset – Optimal PF
Time step #130
(b)
Malaga campus dataset – Optimal PF
Time step #200
10m
(c)
Malaga campus dataset – SM-based PF
Time step #200
(d)

Fig. 6. Experimental results for map building. (a)–(c) Snapshots of the evolution of our optimal PF while building a map for the Málaga campus dataset. (d) The final result for the SM-based particle filter.

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 3(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

$$M_t = O\left(k^{dt}\right), \tag{17}$$

where the factor d needs to be introduced since the above derivation was carried out for just one dimension. This result confirms the intuitive claim above 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 probability distribution function (pdf) for each unidimensional component of the robot pose is a Gaussian, whose variance can be computed by means of a Kalman filter to be $\sigma_t \propto \sigma_0\sqrt{t}$. 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,

$$M_t = O\left(t^{\frac{d}{2}}\right). \tag{18}$$

***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 suboptimality of *all* previous works where the number of particles always remains constant.

6. Experimental Results

In this section we first present experiments where our approach is compared with previous PF algorithms in simulations with a known ground truth, then we apply it to robot localization, and finally we show map building results.

6.1. Simulations With Known Ground Truth

We consider here a one-dimensional linear system with Gaussian transition and observation models. The purpose of using such a simple system is that we are able to contrast the output of the different PFs with the analytical solution from a Kalman filter which provides us with the exact posterior. The situation being simulated is that of an observation model much more peaked than the prior distribution obtained from the system transition model (e.g. in mobile robotics this may represent a poor motion model, such as odometry, and a very precise sensor, such as a laser scanner).

We have compared the following algorithms: a SIR filter with a standard proposal distribution (Rubin 1988), the APF proposed by Pitt and Shephard (1999) and our method (with $B = 100$). Since the APF method does not specify how to estimate the"first stage weights" (Pitt and Shephard 1999) (a term equivalent to Equation (12) in our method), two versions have been put at test: APF-1, using the observation likelihood evaluated at the mean of the system after applying the transition model (one of the recommendations of the authors); and APF-2, using a Monte Carlo approximation exactly as in Equation (12).

The presented simulations consist of executing the different PF methods for a variable number of particles ranging from 100 to 25,000, repeating 200 times each case to

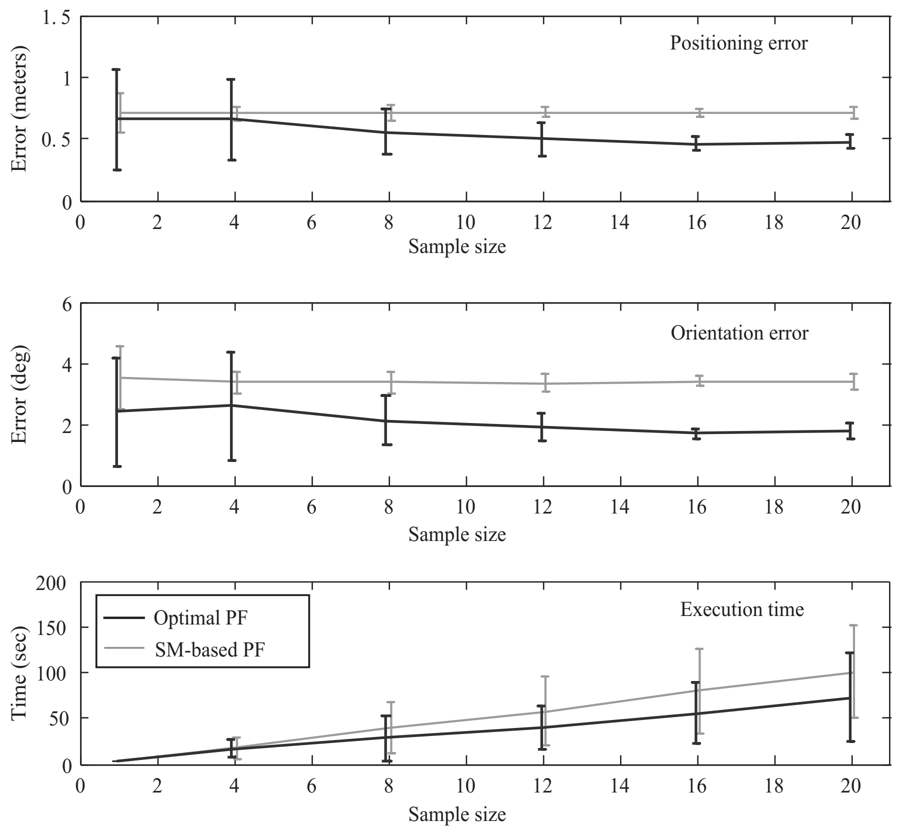


Fig. 7. Errors in localization and orientation. Average execution time for each run (bottom). All graphs have error bars for 95% confidence intervals.

obtain statistically significant results. The resulting particles and their weights for one of the runs are shown in the top row of Figure 4(a) for SIR and our optimal PF. It can be observed how the SIR leads to most of the particles being wasted in non-relevant areas of the state space where they are assigned negligible importance weights. On the other hand, 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 PF, the estimated densities have been reconstructed by means of weighted histograms, displayed in the bottom row of Figure 4(a) along with the analytical solution from the Kalman filter. To evaluate each algorithm, the KLD between the analytical and the estimated distributions has been computed for each sample size (we disabled here the capability of automatically determining the sample size in our method for comparison purposes with the others). The mean KLD values and their confidence intervals for the 200 realizations, summarized in Figure 4(b), confirm that our approach gives estimations closer to the actual posterior (with fewer particles) than previous methods.

One question that may naturally arise in this experiment is regarding the minimum KLD attainable by *any* sample-based representation of the posterior. In fact, we have computed that value by drawing different number of samples from the known, analytical posterior distribution and measuring its KLD to that same posterior. The findings were surprising: the KLD obtained by our optimal PF is almost exactly that lower bound for the KLD, up to an average 0.7% error. In contrast, the mean KLD of the other methods are 540%, 8,200%, and 596% times this lower bound for the SIR, APF-1, and APF-2, respectively. We can conclude that in this experiment, for any sample size, our method provides almost exactly the *best possible* representation of the posterior based on particles.

6.2. Localization

The following localization experiment consists of tracking the pose of a mobile robot equipped with a laser range finder while it is manually guided through an office environment. Concretely, the path described by the robot and the (already built) map of the environment are shown in Figure 5(a). We compare the localization accuracy between the SIR and our optimal PF. The resolution of the occupancy grid is 0.04 m, and the non-parametric observation model is the likelihood field proposed by Thrun (2001); Thrun et al. (2005). In all of the runs the particles were randomly initialized in a small volume of 20 $cm \times 20$ $cm \times 2^{\circ}$ around the real starting pose of the robot.

The accuracy in localization has been evaluated by averaging the localization errors of all of the particles at each time step, and using as the ground truth the robot poses

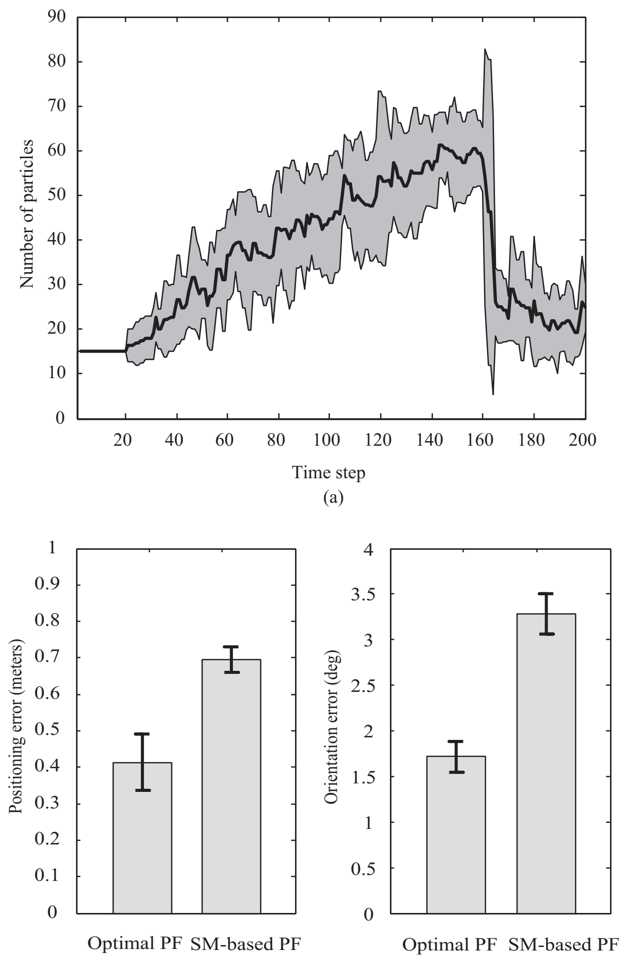


Fig. 8. (a) The dynamic sample size employed by our method, which increases until the loop is closed, around time step 160. The shaded area and the thick plot represent the 95% confidence interval and the mean, respectively, for 20 experiment runs. (b) Errors for our algorithm and SM-based PF for equivalent computational loads. Error bars represent 95% confidence intervals.

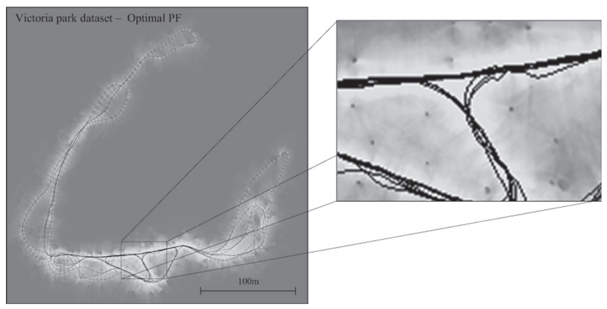


Fig. 9. The map built for the Victoria Park dataset using the optimal PF proposed in the text.

estimated while the map was first built. To obtain significant results we have performed 100 executions for each sample size, ranging from just one particle up to 100. Note that the capability of adapting the sample size in our algorithm has been disabled to provide a fair comparison to a standard SIR (an experiment with a dynamic number of samples can be found in Blanco et al.(2008)). The most interesting conclusion from the results, plotted in Figure 5(b), is that our method has an excellent performance starting from just one particle, whereas a SIR PF needs about 10 particles or more to avoid the filter diverging (e.g. the high average errors for one particle imply that the estimated path is far from the real one). We can also compare the accuracy of each algorithm with respect to the average computational cost instead of the number of samples, as shown in Figure 5(c). In that case, the optimal PF demands almost exactly the same computation time as the SIR to achieve similar levels of accuracy. However, the SIR method needs approximately 20 times more particles than the proposed approach for a given localization accuracy, thus representing an additional storage cost. We must admit that this cost may not be significant for the problem of PF-based pose tracking, but it becomes more important when dealing with global localization.

6.3. SLAM

In this section we illustrate how our optimal filter, as described in Section 5, can be applied to occupancy grid mapping. For comparison, our approach is contrasted to a recent proposal (Grisetti et al. 2007a) which approximates the observation model by a Gaussian through scan matching (SM): we will refer to this approach as SM-based PF. The non-parametric observation model employed here is the same than the localization experiments above, that is, the likelihood field.

The sensory data used as the input to the filters are a part of the Málaga University campus dataset[3], where our mobile robot SENA (Gonzalez et al. 2006) closes a loop of about 60 me in a semi-outdoor scenario, moving around one building. Snapshots of a typical execution for our optimal PF with a dynamic sample size can be seen in Figure 6(a)–(c) at different time steps. The final map obtained for a typical run of the SM-based filter with a fixed sample size of 15 is also represented in Figure 6(d). All of these maps are the most likely grids at each time step, overlapped with all of the robot path hypotheses in the filter.

We have carried out two different experiments to compare the performance of the two alternative PF algorithms. First, the errors in the estimated robot path have been evaluated at the end of the map building process (i.e. after closing the loop). In order to have an approximate ground truth to compare with, a few key robot poses where computed by means of a scan matching algorithm applied to the first robot pose and to those after closing the loop. These key poses where selected by hand by only keeping those laser scans with an accurate alignment to each other, which was verified by human inspection. Next, both PF methods were executed 20 times for different sample sizes, and their positioning and orientation errors measured obtaining the results shown in Figure 7. It can be seen how the newly proposed method outperforms the SM-based PF in localization accuracy for all sample sizes. Moreover, our method also exhibits shorter execution times, although we must remark that there might still be room for improvement in the implementation of both methods. In particular, our optimal PF will remain being faster than the SM-based PF (for the same number of particles) as long as evaluating the observation model $B + R$ times (recall Section 4) is less expensive than running the iterative scan-matching procedure.

In the second experiment, we allowed our method to employ the dynamic sample size, leading to the population sizes depicted in Figure 8(a), where the effects of loop closing are clearly visible around steps 140–160. Then, we determined the number of particles in SM-based filtering that made both approaches equivalent in overall computational times, in order to provide a fair comparison of their localization accuracy. As can be seen in Figure 8(b), our proposal achieves a better performance in this case as well.

Finally, we also briefly discuss a demonstration of map building by our method for an outdoor scenario, in particular, using the standard dataset gathered at Sydney's Victoria Park (Guivant and Nebot 2001). The most likely grid map at the end is shown in Figure 4a, along with a detailed view of the central part of the environment, where the trees can be clearly appreciated. To the best of the authors' 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, data association, and exploits all of the information in the laser scans, whereas in a EKF-like approach only the trees could be used to localize the robot.

7. Conclusions

In this work we have reviewed different Bayesian filtering algorithms, stressing their applications to mobile robots. We have identified situations (localization and SLAM with occupancy grid maps) that require a PF implementation and where optimal filtering was not directly applicable. We have introduced a new algorithm that allows us to apply optimal filtering to those dynamic systems with non-parametric observation models by means of simulations based on rejection sampling and an adaptive sample size. The method is able to focus the samples in the relevant areas of the state space better than previous PF algorithms, which has been confirmed experimentally by successfully tracking the robot pose even with just one particle. This work has also shown how the application of the proposed method to SLAM leads to a filter that increases the number of particles as the robot traverses long loops, and reduces it after closing them, obtaining more accurate maps than previous scan-matching-based approximations. Apart from its demonstrated applications to mobile robotics, the presented algorithm has numerous potential applications to any estimation problem where the lack of an analytical probabilistic model of observations prevented the use of optimal filtering.

Notes

1. For the sake of readability we denote sequences of variables over time using the last time step in the sequence as a superscript.
2. An algorithm such as the C++ STL function random shuffle should be used, which ensures a uniform distribution over the $N!$ possible permutations of an N-vector (see Section 3.4.2 of Knuth (1981)).
3. This dataset is available online: http://babel.isa.uma.es/mrpt/downloads/

Acknowledgment

This work was partially supported by the Spanish FPU grant program and the national research contract CICYT-DPI 2008-03527.

References

Arulampalam, M., Maskell, S., Gordon, N., Clapp, T., Sci, D., Organ, T. and Adelaide, S. (2002). A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Transactions on Signal Processing*, **50**(2): 174–188.

Beevers, K. and Huang, W. (2007). Fixed-lag sampling strategies for particle filtering SLAM. *International Conference on Robotics and Automation (ICRA)*.

Blanco, J., Fernández-Madrigal, J. and Gonzalez, J. (2007). A new approach for large-scale localization and mapping: hybrid metric-topological SLAM. *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 2061–2067.

Blanco, J., Gonzalez, J. and Fernández-Madrigal, J. (2008). An optimal filtering algorithm for non-parametric observation models in robot localization. *IEEE International Conference on Robotics and Automation (ICRA'08)*, pp. 461–466.

Bosse, M., Newman, P., Leonard, J. and Teller, S. (2004). Simultaneous localization and map building in large-scale cyclic environments using the Atlas framework. *The International Journal of Robotics Research*, **23**(12): 1113–1139.

Choo, K. and Fleet, D. (2001). People tracking using hybrid Monte Carlo filtering. *Proceedings of IEEE International Conference on Computer Vision*, Vol. 2, pp. 321–328.

Cover, T. and Thomas, J. (1991). *Elements of Information Theory*. New York, Wiley.

Davison, A., Reid, I., Molton, N. and Stasse, O. (2007). MonoSLAM: real-time single camera SLAM. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **29**(6): 1052–1067.

Dissanayake, M., Newman, P., Clark, S., Durrant-Whyte, H., and Csorba, M. (2001). A solution to the simultaneous localization and map building (SLAM) problem. *IEEE Transactions on Robotics and Automation*, **17**(3): 229–241.

Doucet, A., Briers, M. and Senecal, S. (2006). Efficient block sampling strategies for sequential Monte Carlo methods. *Journal of Computational and Graphical Statistics*, **15**(3): 693–711.

Doucet, A., de Freitas, N. and Gordon, N. (2001). *Sequential Monte Carlo Methods in Practice*. Berlin, Springer.

Doucet, A., de Freitas, N., Murphy, K. and Russell, S. (2000a). Rao–Blackwellised particle filtering for dynamic Bayesian networks. *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, pp. 176–183.

Doucet, A., Godsill, S. and Andrieu, C. (2000b). On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and Computing*, **10**(3): 197–208.

Estrada, C., Neira, J. and Tardos, J. (2005). Hierarchical SLAM: real-time accurate mapping of large environments. *IEEE Transactions on Robotics*, **21**(4): 588–596.

Fox, D. (2003). Adapting the sample size in particle filters through KLD-sampling. *International Journal of Robotics Research*, **22**(12): 985–1003.

Fox, D., Burgard, W., Dellaert, F. and Thrun, S. (1999a). Monte Carlo localization: Efficient position estimation for mobile robots. *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pp. 113–114.

Fox, D., Burgard, W. and Thrun, S. (1999b). Markov localization for mobile robots in dynamic environments. *Journal of Artificial Intelligence Research*, **11**(3): 391–427.

Gonzalez, J., Muñoz, A., Galindo, C., Fernández-Madrigal, J. and Blanco, J. (2006). A description of the SENA robotic wheelchair. *Proceedings of the IEEE Mediterranean Electrotechnical Conference*, pp. 437–440.

Gordon, N., Salmond, D. and Smith, A. (1993). Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEEE Proceedings on Radar and Signal Processing*, **140**(2): 107–113.

Grisetti, G., Stachniss, C. and Burgard, W. (2007a). Improved techniques for grid mapping with Rao–Blackwellized particle filters. *IEEE Transactions on Robotics*, **23**: 34–46.

Grisetti, G., Tipaldi, G., Stachniss, C., Burgard, W. and Nardi, D. (2007b). Fast and accurate slam with Rao–Blackwellized particle filters. *Robotics and Autonomous Systems*, **55**(1): 30–38.

Guivant, J. and Nebot, E. (2001). Optimization of the simultaneous localization and map-building algorithm for real-time implementation. *IEEE Transactions on Robotics and Automation*, **17**(3): 242–257.

Gutmann, J. and Konolige, K. (1999). Incremental mapping of large cyclic environments. *Proceedings of IEEE International Symposium on Computational Intelligence in Robotics and Automation*, pp. 318–325.

Hahnel, D., Burgard, W., Fox, D. and Thrun, S. (2003). An efficient FastSLAM algorithm for generating maps of large-scale cyclic environments from raw laser range measurements. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vol. 1.

Julier, S. (2002) The scaled unscented transformation. *Proceedings of the American Control Conference*, Vol. 6, pp. 4555–4559.

Julier, S. and Uhlmann, J. (1997). A new extension of the Kalman filter to nonlinear systems. *International Symposium of Aerospace/Defense Sensing, Simulation and Controls* pp. 182–193.

Kalman, R. (1960). A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, **82**(1): 35–45.

Knuth, D. (1981). *The Art of Computer Programming: Seminumerical Algorithms*, Vol. 2. Reading, MA, Addison-Wesley.

Liu, J. (1996). Metropolized independent sampling with comparisons to rejection sampling and importance sampling. *Statistics and Computing*, **6**(2): 113–119.

Liu, J. and Chen, R. (1998). Sequential Monte Carlo Methods for dynamic systems. *Journal of the American Statistical Association*, **93**(443): 1032–1044.

Loutfi, A., Lilienthal, A., Blanco, J., Galindo, C. and Gonzalez, J. (2007). Integrating SLAM into gas distribution mapping. *Proceedings of the IEEE International Conference on Robotics and Automation*.

Montemerlo, M. (2003). *FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem With Unknown Data Association*. PhD Thesis, University of Washington.

Montemerlo, M., Thrun, S., Koller, D. and Wegbreit, B. (2002a). FastSLAM: a factored solution to the simultaneous localization and mapping problem. *Proceedings of the AAAI National Conference on Artificial Intelligence*, pp. 593–598.

Montemerlo, M., Thrun, S. and Whittaker, W. (2002b). Conditional particle filters for simultaneous mobile robot localization and people-tracking. *Proceedings of the IEEE International Conference on Robotics and Automation*, Vol. 1, pp. 695–701.

Moravec, H. and Elfes, A. (1985). High resolution maps from wide angle sonar. *Proceedings of the IEEE International Conference on Robotics and Automation*, Vol. 2.

Murphy, K. (1999). Bayesian map learning in dynamic environments. *Advances in Neural Information Processing Systems (NIPS)*, **12**: 1015–1021.

Neira, J. and Tardós, J. (2001). Data association in stochastic mapping using the Joint Compatibility test. *IEEE Transactions on Robotics and Automation*, **17**(6): 890–897.

Nummiaro, K., Koller-Meier, E. and Van Gool, L. (2003). An adaptive color-based particle filter. *Image and Vision Computing*, **21**(1): 99–110.

Okuma, K., Taleghani, A., de Freitas, N., Little, J. and Lowe, D. (2004). A boosted particle filter: Multitarget detection and tracking. *European Conference on Computer Vision*, **1**: 28–39.

Pitt, M. and Shephard, N. (1999). Filtering via simulation: auxiliary particle filters. *Journal of the American Statistical Association*, **94**(446): 590–591.

Ranganathan, A., Menegatti, E. and Dellaert, F. (2006). Bayesian inference in the space of topological maps. *IEEE Transactions on Robotics*, **22**(1): 92–107.

Ristic, B., Arulampalam, S. and Gordon, N. (2004). *Beyond the Kalman Filter: Particle Filters for Tracking Applications*. Norwood, MA, Artech House.

Rubin, D. (1987). 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*, **82**(398): 543–546.

Rubin, D. (1988). Using the SIR algorithm to simulate posterior distributions. *Bayesian Statistics*, **3**: 395–402.

Schulz, D., Burgard, W., Fox, D. and Cremers, A. (2001). Tracking multiple moving targets with a mobile robot using particle filters and statistical data association. *Proceedings of the IEEE International Conference on Robotics and Automation*, Vol. 2.

Tenne, D. and Singh, T. (2003). The higher order unscented filter. *Proceedings of the American Control Conference*, Vol. 3, pp. 2441–2446.

Thrun, S. (2001). A probabilistic on-line mapping algorithm for teams of mobile robots. *The International Journal of Robotics Research*, **20**(5): 335.

Thrun, S. (2002). *Robotic Mapping: A Survey*. School of Computer Science, Carnegie Mellon University.

Thrun, S. (2003). Learning occupancy grid maps with forward sensor models. *Autonomous Robots*, **15**(2): 111–127.

Thrun, S., Burgard, W. and Fox, D. (2005). *Probabilistic Robotics*. Cambridge, MA, The MIT Press.

Thrun, S., Fox, D., Burgard, W. and Dellaert, F. (2001). Robust Monte Carlo localization for mobile robots. *Artificial Intelligence*, **128**(1–2): 99–141.

Thrun, S., Liu, Y., Koller, D., Ng, A., Ghahramani, Z. and Durrant-Whyte, H. (2004). Simultaneous localization and mapping with sparse extended information filters. *The International Journal of Robotics Research*, **23**(7–8): 693.

Vlassis, N., Terwijn, B. and Krose, B. (2002). Auxiliary particle filter robot localization from high-dimensional sensor observations. *Proceedings of the IEEE International Conference on Robotics and Automation*, Vol. 1.

Wan, E. and Van Der Merwe, R. (2000). The unscented Kalman filter for nonlinear estimation. *Proceedings of the IEEE Adaptive Systems for Signal Processing, Communications, and Control Symposium*, pp. 153–158.

Appendix: Bernoulli Trials for Accepting a Sample in Rejection Sampling

Consider two independent probability distributions, which we 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,

$$p = \frac{p(z|x)}{\max\{p(z|x)\}}, \tag{19}$$

where $p(z|x)$ denotes the observation likelihood. Since $p(z|x)$ is a Gaussian, the quotient above reduces to

$$p = \exp\left\{-\frac{1}{2}\left(\frac{x-\Delta\mu}{\sigma_o}\right)^2\right\}. \tag{20}$$

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

$$E[p] = \int_{-\infty}^{\infty} \exp\left\{-\frac{1}{2}\left(\frac{x-\Delta\mu}{\sigma_o}\right)^2\right\}\mathcal{N}(x; 0, \sigma_p^2)\,dx. \tag{21}$$

By multiplying and dividing by the appropriate constant, the terms inside the integral can be transformed into the product of two Gaussians:

$$E[p] = \sigma_o\sqrt{2\pi}\int_{-\infty}^{\infty}\mathcal{N}(x; \Delta\mu, \sigma_o^2)\mathcal{N}(x; 0, \sigma_p^2)\,dx. \tag{22}$$

Following a probabilistic interpretation of the integral above, we can consider it the likelihood of two random variables, each following one of the two normal distributions, coinciding at exactly the same point. The reason for the integral is that the point of coincidence can be anywhere on the state space. Once the equation has been considered from this point of view, it is easier to evaluate the integral if we rewrite it in terms of the likelihood, evaluated at the origin, of the new random variable arising from substracting the original two variables. Since these original variables were normally distributed and substraction is a linear operation, we have that the new variable must be also a Gaussian, with a mean of $\Delta\mu$ and variance $\sigma_o^2 + \sigma_p^2$. Then, we proceed rewriting Equation (22) as

$$E[p] = \sigma_o\sqrt{2\pi}\cdot\mathcal{N}(0; \Delta\mu, \sigma_o^2 + \sigma_p^2)$$

$$= \frac{\sigma_o}{\sqrt{\sigma_o^2+\sigma_p^2}}\exp\left(-\frac{1}{2}\frac{\Delta\mu^2}{\sigma_o^2+\sigma_p^2}\right).$$

For convenience, we define the constant $\tau = \sigma_o/\sigma_p$, such as

$$E[p] = \frac{1}{\sqrt{1+\tau^{-2}}}\exp\left(-\frac{1}{2\sigma_p^2}\frac{\Delta\mu^2}{1+\tau^2}\right), \tag{23}$$

which is the final expression used to obtain the plots in Figure 2.