Proceedings of Australasian Conference on Robotics and Automation, 3-5 Dec 2012, Victoria University of Wellington, New Zealand.

Monte Carlo Sampling of Non-Gaussian Proposal Distribution in Feature-Based RBPF-SLAM

Nina Marhamati, Hamid D. Taghirad
Advanced Robotics and Automated Systems
K. N. Toosi University of Technology
Tehran, Iran
nina@siu.edu, taghirad@kntu.ac.ir

Kasra Khosoussi
Centre for Autonomous Systems
Faculty of Engineering and IT
University of Technology, Sydney
Australia
kasra.khosoussi@uts.edu.au

Abstract

Particle filters are widely used in mobile robot localization and mapping. It is well-known that choosing an appropriate proposal distribution plays a crucial role in the success of particle filters. The proposal distribution conditioned on the most recent observation, known as the optimal proposal distribution (OPD), increases the number of effective particles and limits the degeneracy of filter. Conventionally, the OPD is approximated by a Gaussian distribution, which can lead to failure if the true distribution is highly non-Gaussian. In this paper we propose two novel solutions to the problem of feature-based SLAM, through Monte Carlo approximation of the OPD which show superior results in terms of mean squared error (MSE) and number of effective samples. The proposed methods are capable of describing non-Gaussian OPD and dealing with nonlinear models. Simulation and experimental results in large-scale environments show that the new algorithms outperform the aforementioned conventional methods.

1 Introduction

Sequential Monte Carlo (SMC) methods, also known as particle filters, have shown a great success in localizing mobile robots and solving simultaneous localization and mapping (SLAM) problem due to their capability in capturing non-Gaussian and multi-modal posterior distributions and dealing with nonlinear models. Murphy [Murphy, 2000] proposed to make use of Rao-Blackwellized particle filter (RBPF) to solve the mapping problem through conditioning the map on robot trajectory. Hence, it is only required to sample from the robot path, which reduces the dimension of sample-space, while the locations of the features can be estimated analytically and independent of each other given the robot trajectory. The first efficient SLAM algorithm based on RBPF, FastSLAM [Montemerlo *et al.*, 2002], employed a particle filter to estimate the robot pose, and used EKF in order to estimate the position of the landmarks for each particle.

Particle filters suffer from the so-called degeneracy problem which occurs since the variance of importance weights grows with time [Doucet, 1998]. The proposal distribution that minimizes the conditional variance of importance weights is called the optimal proposal distribution (OPD) which is constructed by considering the most recent observation [Doucet, 1998]. This distribution cannot be obtained in closed form for the nonlinear models of SLAM. Local linearization [Doucet, 1998] approximates the OPD as a Gaussian by linearizing the observation model using the first-order Taylor expansion. Montemerlo *et al.* proposed FastSLAM 2.0 [Montemerlo *et al.*, 2003], which uses local linearization in order to sample from the OPD. The results showed major improvements compared to FastSLAM 1.0. However, FastSLAM 2.0 is subjected to linearization errors which can lead to the filter divergence. Furthermore, its inability to describe non-Gaussian proposal distributions, is another important disadvantage.

Merwe *et al.* introduced unscented particle filter (UPF) [Van Der Merwe *et al.*, 2001] in which instead of linearizing nonlinear models, unscented Kalman filter (UKF) is used in order to obtain the mean and covariance of OPD and to approximate it by a Gaussian distribution. UPF yields more accurate estimates compared to local linearization. Kim *et al.* have recently applied UPF to the problem of feature-based SLAM [Kim *et al.*, 2008]. Their method, Unscented FastSLAM (UFastSLAM), has shown superior results compared to FastSLAM 2.0 in terms of the MSE and consistency. More recently, Saha *et al.* have proposed a more accurate Gaussian approximation algorithm for the OPD through exact moment matching (EMM) [Saha *et al.*, 2009]. Through statistical analyses and experiments, Stachniss *et al.* [Stachniss *et al.*, 2007] have shown that Gaussian approximation fails to estimate the map accurately in certain cases. Their work shows that the ability to describe multi-modal proposal distributions is necessary to prevent such failures. In other words, Gaussian approximation of OPD in SLAM is doomed to failure whenever the OPD is highly non-Gaussian and multi-modal, no matter how accurate the approximation method is.

Monte Carlo (MC) sampling methods are capable of sampling from arbitrary distributions. Recently, Blanco *et al.* [Blanco *et al.*, 2010] proposed a method to sample from the OPD using rejection sampling (RS), for the case of non-parametric observation models and grid-based maps. The basic idea of their method was presented in [Doucet, 1998] and [Liu and Chen, 1998]. Their algorithm addressed the grid-based form of the SLAM problem and it yielded better results compared to [Grisetti *et al.*, 2007] – a grid-based RBPF-SLAM algorithm which employs Gaussian approximation for the OPD. Nevertheless, their method has a stochastic time-complexity which can be considered as a major disadvantage.

In this work we propose to approximate the OPD using local MC (LMC) sampling methods. We propose LMC-1 and LMC-2, two novel approaches to the problem of feature-based SLAM based on this idea. The proposed algorithms employ RS and importance sampling (IS) in order to generate samples according to the OPD. Hence, our algorithms are perfectly capable of sampling from the non-Gaussian and multi-modal OPD of SLAM. Unlike [Blanco *et al.*, 2010], the proposed algorithms are focused on the feature-based form of the SLAM problem. Moreover, they do not suffer from the stochastic time-complexity.

2 Problem Formulation

In this section we present a formulation of the feature-based SLAM problem. The state variable at time $t \in \mathbb{N}$ is represented by $\mathbf{x_t} \in \mathbb{R}^{n_x}$ which consists of robot pose $\mathbf{s_t} \in \mathbb{R}^{n_s}$ and the map Θ. The map is represented by a set of static landmarks $\{\theta_\mathbf{i}; \mathbf{i} = \mathbf{1}, \dots, \mathbf{L}\}$ where $\theta_\mathbf{i} \in \mathbb{R}^{n_\theta}$. The robot pose process is modelled as a Markov process with transition equation $p(\mathbf{s_t}|\mathbf{s_{t-1}})$ (known as the motion model). The observations $\mathbf{z_t} \in \mathbb{R}^{n_z}$ are conditionally independent given the state process with marginal distribution $p(\mathbf{z_t}|\mathbf{x_t})$ known as the observation model. We denote the set of any variable like c_i up to time t by $c_{1:t} \triangleq \{c_1, \dots, c_t\}$.

This model is associated with the following state space equations

$$\mathbf{s_t} = \mathbf{h}(\mathbf{s_{t-1}}, \mathbf{u_t}) + \mathbf{v_t} \tag{1}$$

$$\mathbf{z_t} = \mathbf{g}(\mathbf{s_t}, \theta_{\mathbf{n_t}}) + \mathbf{r_t} \tag{2}$$

where $h(\cdot,\cdot)$ and $g(\cdot,\cdot)$ are nonlinear functions describing the robot motion and measurement model respectively, $\mathbf{v_t}$ and $\mathbf{r_t}$ are zero mean white Gaussian noises with covariances $\mathbf{Q_t}$ and $\mathbf{R_t}$, $\mathbf{u_t}$ is the control input, and $\theta_{\mathbf{n_t}}$ is the observed feature at time t. Hence we have

$$p(\mathbf{s_t}|\mathbf{s_{t-1}}, \mathbf{u_t}) = \mathcal{N}(\mathbf{s_t}; \mathbf{h}(\mathbf{s_{t-1}}, \mathbf{u_t}), \mathbf{Q_t}) \tag{3}$$

$$p(\mathbf{z_t}|\mathbf{s_t}, \theta_{\mathbf{n_t}}) = \mathcal{N}(\mathbf{z_t}; \mathbf{g}(\mathbf{s_t}, \theta_{\mathbf{n_t}}), \mathbf{R_t}) \tag{4}$$

In this work, we consider the data association parameter n_t as known. Also from now on, we omit the control input $\mathbf{u_t}$ for notational convenience.

Our goal is to estimate the robot pose and the map for any time step t using the observations and control inputs available up to that time. More precisely, we are interested in estimating the filtering distribution $p(\mathbf{x_t}|\mathbf{z_{1:t}})$ and the expectation

$$\mathbb{E}[\mathbf{x_t}|\mathbf{z_{1:t}}] = \int \mathbf{x_t p(x_t|z_{1:t})dx_t} \tag{5}$$

as the minimum mean square error (MMSE) estimate, sequentially in time.[1]

3 Sequential Monte Carlo Methods in SLAM

RBPF solutions to the SLAM problem use the following factorization of the full SLAM posterior [Murphy, 2000]

$$p(\mathbf{s_{0:t}}, \mathbf{\Theta}|\mathbf{z_{1:t}}) = \mathbf{p}(\mathbf{s_{0:t}}|\mathbf{z_{1:t}}) \prod_{\mathbf{i=1}}^{\mathbf{L}} \mathbf{p}(\theta_\mathbf{i}|\mathbf{s_{0:t}}, \mathbf{z_{1:t}})$$

where $p(\theta_\mathbf{i}|\mathbf{s_{0:t}}, \mathbf{z_{1:t}})$ can be computed analytically by a low dimensional $(n_\theta \times n_\theta)$ EKF. Therefore we can employ SMC methods only to estimate $p(\mathbf{s_{0:t}}|\mathbf{z_{1:t}})$, which will reduce the variance of filter, along with the computational costs. If we were able to sample N independent and identically distributed (i.i.d.) samples $\{\mathbf{s_{0:t}^{[i]}}; \mathbf{i} = \mathbf{1}, \dots, \mathbf{N}\}$

[1]In fact in particle filtering, the filtering distribution is obtained indirectly by marginalizing the distribution $p(\mathbf{x_{0:t}}|\mathbf{z_{1:t}})$ which is known as the full SLAM posterior.

from the path posterior, the posterior could be approximated by

$$\hat{p}^*(\mathbf{s_{0:t}}|\mathbf{z_{1:t}}) = \frac{1}{N}\sum_{i=1}^{N}\delta_{\mathbf{s^{[i]}_{0:t}}}(\mathbf{s_{0:t}}) \tag{6}$$

in which $\delta_{\mathbf{s^{[i]}_{0:t}}}(\mathbf{s_{0:t}})$ denotes the Dirac delta mass located at $\mathbf{s^{[i]}_{0:t}}$. Using (6), SLAM posterior can be obtained as

$$\frac{1}{N}\sum_{i=1}^{N}[\prod_{j=1}^{L} p(\theta_{\mathbf{j}}|\mathbf{s^{[i]}_{0:t}},\mathbf{z_{1:t}})]\delta_{\mathbf{s^{[i]}_{0:t}}}(\mathbf{s_{0:t}})$$

which can be interpreted as a set of N particles describing the robot path, each accompanied by L Gaussian distributions representing the map posterior for that particle [Montemerlo *et al.*, 2002].

The strong law of large numbers implies that any expectation computed with respect to (6) converges almost surely to its true value as N goes to infinity. Nevertheless, the path posterior is only known up to a constant, and therefore, samples cannot be generated directly from $p(\mathbf{s_{0:t}}|\mathbf{z_{1:t}})$. Instead, we can sample according to an easy to sample distribution such as $\pi(\mathbf{s_{0:t}}|\mathbf{z_{1:t}})$, called importance function, and assign weights to these generated samples proportional to $\frac{p(\mathbf{s_{0:t}}|\mathbf{z_{1:t}})}{\pi(\mathbf{s_{0:t}}|\mathbf{z_{1:t}})}$ in order to approximate (6). Sampling according to $\pi(\mathbf{s_{0:t}}|\mathbf{z_{1:t}})$ from scratch, implies a growing computational complexity over time. By choosing importance functions of the form

$$\pi(\mathbf{s_{0:t}}|\mathbf{z_{1:t}}) = \pi(\mathbf{s_{0:t-1}}|\mathbf{z_{1:t-1}})\pi(\mathbf{s_t}|\mathbf{s_{0:t-1}},\mathbf{z_{1:t}}) \tag{7}$$

one is able to reuse the previous generated samples $\mathbf{s^{[i]}_{0:t-1}}$ and construct $\mathbf{s^{[i]}_{0:t}}$ as

$$\mathbf{s^{[i]}_{0:t}} = \langle \mathbf{s^{[i]}_{0:t-1}}, \mathbf{s^{[i]}_{t}} \rangle$$

where $\mathbf{s^{[i]}_{t}}$ is sampled from $\pi(\mathbf{s_t}|\mathbf{s^{[i]}_{0:t-1}},\mathbf{z_{1:t}})$. Using (7), importance weights can be computed up to a normalizing constant as [2]

$$w^{[i]}_{0:t} = w^{[i]}_{0:t-1}\frac{p(\mathbf{z_t}|\mathbf{s^{[i]}_t})\mathbf{p}(\mathbf{s^{[i]}_t}|\mathbf{s^{[i]}_{t-1}})}{\pi(\mathbf{s^{[i]}_t}|\mathbf{s^{[i]}_{0:t-1}},\mathbf{z_{1:t}})} \propto \frac{p(\mathbf{s^{[i]}_{0:t}}|\mathbf{z_{1:t}})}{\pi(\mathbf{s^{[i]}_{0:t}}|\mathbf{z_{1:t}})} \tag{8}$$

Finally, path posterior can be approximated by

$$\hat{p}(\mathbf{s_{0:t}}|\mathbf{z_{1:t}}) = \sum_{i=1}^{N}\tilde{w}^{[i]}_{0:t}\delta_{\mathbf{s^{[i]}_{0:t}}}(\mathbf{s_{0:t}})$$

where

$$\tilde{w}^{[i]}_{0:t} = \frac{w^{[i]}_{0:t}}{\sum_{j=1}^{N} w^{[j]}_{0:t}}$$

[2] With a slight abuse of notation, we use $w^{[i]}_{0:t}$ to emphasize that this weight is associated with the path sample $s^{[i]}_{0:t}$.

In this way, MMSE estimate (5) is obtained by computing the weighted mean of these samples. This method is known as sequential importance sampling (SIS). It is proved that the variance of importance weights grows with time [Kong *et al.*, 1994] which leads to the so-called "degeneracy" problem in which after a few steps, all but one of the particles will have very insignificant normalized weights. This problem was addressed in "bootstrap filter" [Gordon *et al.*, 1993] by introducing resampling steps at the end of each SIS iteration, a process in which particles with low normalized weights are eliminated and those with high normalized weights are copied. See [Douc and Cappé, 2005] for a review on different resampling schemes. Effective sample size (ESS) [Kong *et al.*, 1994], which is defined as

$$N_{\text{eff}} = \frac{1}{\sum_{i=1}^{N}\tilde{w}^{[i]2}_{0:t}} \tag{9}$$

is a measure of the degeneracy of SIS that can be used to avoid unnecessary resampling steps. Resampling is performed only if N_{eff} is below a threshold N_{thr}, typically $N_{thr} = N/2$.

Selecting an appropriate proposal distribution $\pi(\mathbf{s_t}|\mathbf{s_{0:t-1}},\mathbf{z_{1:t}})$ is of utmost importance to the success of SIS. Various proposal distributions have been proposed and evaluated for many applications [Doucet *et al.*, 2001]. FastSLAM 1.0 uses the simplest and most common choice, the transition density $p(\mathbf{s_t}|\mathbf{s_{t-1}})$, which according to (3) is a known Gaussian. As a consequence, (8) will be simplified as

$$w^{[i]}_{0:t} = w^{[i]}_{0:t-1}p(\mathbf{z_t}|\mathbf{s^{[i]}_t}) \tag{10}$$

in which $p(\mathbf{z_t}|\mathbf{s^{[i]}_t})$ can be obtained in closed form by linearizing $g(\cdot,\cdot)$ with respect to the observed feature.

In FastSLAM 1.0, new samples are generated without any knowledge of the most recent observation $\mathbf{z_t}$, and therefore a great number of particles may be wasted in the less important regions of state-space. Consequently, those particles will receive small weights and become discarded in the resampling stage and only a few number of particles would be duplicated for many times, which is obviously undesirable. This phenomenal has a more disastrous effect in a case when the observation model being used is more accurate than the motion model (e.g. accurate laser range finders and noisy encoder).

It is proved that by choosing $p(\mathbf{s_t}|\mathbf{s_{t-1}},\mathbf{z_t})$ as the proposal distribution, the variance of unnormalized importance weights conditional upon $\mathbf{s^{[i]}_{0:t-1}}$ and $\mathbf{z_{0:t}}$ is minimized [Doucet, 1998]. This distribution is called the optimal proposal distribution (OPD) and can be used in order to limit the degeneracy of the filter. In this case, importance weights (8) can be obtained as

$$w^{[i]}_{0:t} = w^{[i]}_{0:t-1}p(\mathbf{z_t}|\mathbf{s^{[i]}_{t-1}}) \tag{11}$$

In general, neither $p(\mathbf{s_t}|\mathbf{s_{t-1}}, \mathbf{z_t})$ nor $p(\mathbf{z_t}|\mathbf{s_{t-1}})$ can be computed in closed form. Nevertheless, if the observation equation (2) was linear, these distributions would become available as Gaussian distributions. The so-called local linearization method approximates these distributions by linearizing $g(\cdot,\cdot)$ using the first-order Taylor expansion [Doucet, 1998]. FastSLAM 2.0 employs local linearization method to generate samples according to the OPD as follows.
The OPD can be expanded using the Bayes rule as

$$p(\mathbf{s_t}|\mathbf{s^{[i]}_{t-1}}, \mathbf{z_t}) \propto \mathbf{p(z_t|s_t)p(s_t|s^{[i]}_{t-1})} \qquad (12)$$

The distribution $p(\mathbf{z_t}|\mathbf{s_t})$ can be written in terms of the SLAM observation model as

$$\int p(\mathbf{z_t}|\mathbf{s_t}, \theta_{\mathbf{n_t}})\mathbf{p}(\theta_{\mathbf{n_t}}|\mathbf{s^{[i]}_{0:t-1}}, \mathbf{z_{0:t-1}})\mathbf{d}\theta_{\mathbf{n_t}} \qquad (13)$$

in which $p(\theta_{\mathbf{n_t}}|\mathbf{s^{[i]}_{0:t-1}}, \mathbf{z_{0:t-1}})$ represents the landmark estimate at time $t-1$ which is distributed according to $\mathcal{N}(\theta_{\mathbf{n_t}}; \mu^{[\mathbf{i}]}_{\mathbf{n_t,t-1}}, \mathbf{\Sigma}^{[\mathbf{i}]}_{\mathbf{n_t,t-1}})$. A first-order Taylor expansion of $g(\cdot,\cdot)$ yields to [Montemerlo *et al.*, 2003]

$$g(\mathbf{s_t}, \theta_{\mathbf{n_t}}) \approx \hat{\mathbf{z}}^{[\mathbf{i}]}_{\mathbf{t}} + \nabla\mathbf{g}_{\theta_{\mathbf{n_t}}}(\theta_{\mathbf{n_t}} - \mu^{[\mathbf{i}]}_{\mathbf{n_t,t-1}}) + \nabla\mathbf{g}_{\mathbf{s_t}}(\mathbf{s_t} - \hat{\mathbf{s}}^{[\mathbf{i}]}_{\mathbf{t}}) \qquad (14)$$

where

$$\hat{\mathbf{s}}^{[i]}_t = h(\mathbf{s^{[i]}_{t-1}}, \mathbf{u_t}) \quad , \quad \hat{\mathbf{z}}^{[i]}_t = g(\hat{\mathbf{s}}^{[i]}_t, \mu^{[\mathbf{i}]}_{\mathbf{n_t,t-1}})$$

and

$$\nabla\mathbf{g}_{\mathbf{s_t}} = \left.\frac{\partial\mathbf{g}}{\partial\mathbf{s_t}}\right|_{(\hat{\mathbf{s}}^{[i]}_{\mathbf{t}}, \mu^{[\mathbf{i}]}_{\mathbf{n_t,t-1}})} \quad , \quad \nabla\mathbf{g}_{\theta_{\mathbf{n_t}}} = \left.\frac{\partial\mathbf{g}}{\partial\theta_{\mathbf{n_t}}}\right|_{(\hat{\mathbf{s}}^{[i]}_{\mathbf{t}}, \mu^{[\mathbf{i}]}_{\mathbf{n_t,t-1}})}$$

Using (12)-(14), FastSLAM 2.0 approximates the OPD as $\mathcal{N}(\mathbf{s_t}; \mu^{[\mathbf{i}]}_{\mathbf{s_t}}, \mathbf{\Sigma}^{[\mathbf{i}]}_{\mathbf{s_t}})$ where

$$\Sigma^{[i]}_{\mathbf{s_t}} = [\nabla\mathbf{g}^{\mathbf{T}}_{\mathbf{s_t}}(\mathbf{R_t} + \nabla\mathbf{g}_{\theta_{\mathbf{n_t}}}\mathbf{\Sigma}^{[\mathbf{i}]}_{\mathbf{n_t,t-1}}\nabla\mathbf{g}^{\mathbf{T}}_{\theta_{\mathbf{n_t}}})^{-\mathbf{1}}\nabla\mathbf{g}_{\mathbf{s_t}} + \mathbf{Q}^{-\mathbf{1}}_{\mathbf{t}}]^{-\mathbf{1}}$$
$$\mu^{[i]}_{\mathbf{s_t}} = \Sigma^{[i]}_{\mathbf{s_t}}\nabla\mathbf{g}^{\mathbf{T}}_{\mathbf{s_t}}(\mathbf{R_t} + \nabla\mathbf{g}_{\theta_{\mathbf{n_t}}}\mathbf{\Sigma}^{[\mathbf{i}]}_{\mathbf{n_t,t-1}}\nabla\mathbf{g}^{\mathbf{T}}_{\theta_{\mathbf{n_t}}})^{-1}(\mathbf{z_t} - \hat{\mathbf{z}}^{[i]}_{\mathbf{t}}) + \hat{\mathbf{s}}^{[i]}_t$$

FastSLAM performs resampling at the end of each step. Therefore (11) is reduced to $p(\mathbf{z_t}|\mathbf{s^{[i]}_{t-1}})$ which can be approximated using the same linearization used above as

$$w^{[i]}_{0:t} \approx \mathcal{N}(\mathbf{z_t}; \hat{\mathbf{z}}^{[\mathbf{i}]}_{\mathbf{t}}, \nabla\mathbf{g}_{\mathbf{s_t}}\mathbf{Q_t}\nabla\mathbf{g}^{\mathbf{T}}_{\mathbf{s_t}} + \nabla\mathbf{g}_{\theta_{\mathbf{n_t}}}\mathbf{\Sigma}^{[\mathbf{i}]}_{\mathbf{n_t,t-1}}\nabla\mathbf{g}^{\mathbf{T}}_{\theta_{\mathbf{n_t}}} + \mathbf{R_t})$$

As it was mentioned earlier, FastSLAM 2.0 suffers from two major drawbacks regarding the approximation of the OPD. The Gaussian approximation is unable to describe cases were the true OPD is multi-modal (see [Stachniss *et al.*, 2007] for more details). Moreover, the linearization error can lead to filter divergence and inconsistent estimates, as reported in [Bailey *et al.*, 2006]. In the following sections we present two alternative methods in order to address these issues.

4 MC approximation of the optimal proposal distribution

The basic idea of this work is to construct the OPD using MC sampling methods. Therefore, the OPD will be described by a set of random samples which is in contrast to the existing methods such as [Montemerlo *et al.*, 2003], [Kim *et al.*, 2008], [Van Der Merwe *et al.*, 2001], [Grisetti *et al.*, 2007] and [Saha *et al.*, 2009] in which an analytic approximation (i.e. Gaussian) is provided for the OPD. The OPD in SLAM can be highly non-Gaussian and multi-modal [Stachniss *et al.*, 2007]. As a consequence, Gaussian approximation, even in perfect conditions, fails to describe the OPD adequately. The ability to draw samples from the various modes of the OPD is crucial to the success of the particle filter because eventually these samples will be used in order to estimate the target distribution. Gaussian approximation is, by definition, incapable of describing the various modes of the OPD.

In this work, the OPD $p(\mathbf{s_t}|\mathbf{s^{[i]}_{t-1}}, \mathbf{z_t})$, is considered as a local target distribution and is estimated using M samples, initially drawn from another distribution such as $q(\mathbf{s_t}|\mathbf{s^{[i]}_{t-1}}, \mathbf{z_t})$ – the so-called local proposal distribution. An overview of the procedure will be as follows. First, temporary samples are initially generated according to this local proposal distribution. Next, this set of temporary samples will be transformed by means of MC sampling methods into another set of samples distributed according to the OPD. At this stage, we would have a set of particles which describes the OPD. Finally, in order to reach the main target distribution (i.e. robot path posterior), importance weights will be assigned to these samples according to SIS (11) and resampling will be performed if it is necessary. It is of great importance to note that unlike SIS, these local MC methods are employed on a fixed low dimensional space $\mathbb{R}^{n_s}$ and therefore are immune to such difficulties mentioned for SIS.

The most important advantage of our method over FastSLAM 2.0, UFastSLAM and EMM is its ability to cope with and sample from non-Gaussian (e.g. multi-modal) OPD in nonlinear problems such as SLAM. Furthermore, it is clear that by increasing M, the OPD is obtained more accurately.[3] Therefore, by controlling the number of local samples, M, one can reach the desired accuracy based on the available computational resources at the moment which is of great importance in online applications. This is in contrast to FastSLAM 2.0 where the linearization error can be neither avoided nor controlled.

[3]Here we are interpreting these local MC steps as MC sampling rather than MC integration methods. Nevertheless, samples generated in these steps are further being used to compute the desired expectation (5).

In the following sections, we present two novel solutions to the feature-based SLAM problem, namely LMC-1 and LMC-2. LMC-1 employs rejection sampling (RS) [Neumann, 1951], while LMC-2 uses importance sampling in order to sample from the OPD. In both of these approaches, the process of updating landmark estimates is identical to FastSLAM [Montemerlo *et al.*, 2002], and therefore is not presented here.

4.1 LMC-1

Rejection sampling (RS) can be used in order to obtain samples from the OPD by accepting or rejecting samples generated according to $q(\mathbf{s_t}|\mathbf{s_{t-1}^{[i]}},\mathbf{z_t})$. Suppose there is a constant C_i such that $\forall \mathbf{s_t}$ we have $C_i q(\mathbf{s_t}|\mathbf{s_{t-1}^{[i]}},\mathbf{z_t}) \geq \mathbf{p}(\mathbf{s_t}|\mathbf{s_{t-1}^{[i]}},\mathbf{z_t})$, then each RS step may consist of [Neumann, 1951]

- Generate $\mathbf{s_t^{[i,j]}}$ according to $q(\mathbf{s_t}|\mathbf{s_{t-1}^{[i]}},\mathbf{z_t})$
- Generate $u \sim \mathcal{U}[0,1]$
- Compute $\Delta_{i,j}$:

$$\Delta_{i,j} = \frac{p(\mathbf{s_t^{[i,j]}}|\mathbf{s_{t-1}^{[i]}},\mathbf{z_t})}{C_i q(\mathbf{s_t^{[i,j]}}|\mathbf{s_{t-1}^{[i]}},\mathbf{z_t})} \;(\leq 1) \qquad (15)$$

- Accept $\mathbf{s_t^{[i,j]}}$ if $u < \Delta_{i,j}$

Therefore, the accepted samples are distributed according to $p(\mathbf{s_t}|\mathbf{s_{t-1}^{[i]}},\mathbf{z_t})$. A proper choice for $q(\mathbf{s_t}|\mathbf{s_{t-1}^{[i]}},\mathbf{z_t})$ is the SLAM motion model $p(\mathbf{s_t}|\mathbf{s_{t-1}^{[i]}})$. Therefore, according to (12) and (15) $\Delta_{i,j}$ can be obtained as

$$\Delta_{i,j} = \frac{1}{C_i} p(\mathbf{z_t}|\mathbf{s_t^{[i,j]}})$$

To meet the condition depicted earlier, C_i can be chosen as $\max p(\mathbf{z_t}|\mathbf{s_t^{[i,j]}})$. In practice we use a modified version of RS, in which $\hat{C}_i = \max_j p(\mathbf{z_t}|\mathbf{s_t^{[i,j]}})$ is being used instead of C_i [Caffo *et al.*, 2002].

Consequently, we need to evaluate $p(\mathbf{z_t}|\mathbf{s_t^{[i,j]}})$, which according to (13) can be done using the first-order Taylor expansion of $g(\cdot,\cdot)$ with respect to the observed feature as [4]

$$\mathcal{N}(\mathbf{z_t}; \hat{\mathbf{z}}_\mathbf{t}^{[\mathbf{i}]}, \nabla \mathbf{g}_{\theta_{\mathbf{n_t}}} \mathbf{\Sigma}_{\mathbf{n_t},\mathbf{t-1}}^{[\mathbf{i}]} \nabla \mathbf{g}_{\theta_{\mathbf{n_t}}}^{\mathbf{T}} + \mathbf{R_t}) \qquad (16)$$

Hence an iteration of the algorithm proceeds as follows: for each particle $\mathbf{s_{t-1}^{[i]}}$ $(i = 1,\ldots,N)$, M samples are drawn from $p(\mathbf{s_t}|\mathbf{s_{t-1}^{[i]}})$. This can be done by sampling M i.i.d. samples from $p(\mathbf{v_t})$ and passing them along with $\mathbf{s_{t-1}^{[i]}}$ and $\mathbf{u_t}$ through the motion equation (1). Then by applying the procedure described above, some of these NM generated samples are accepted. Thus we have

$$\hat{p}(\mathbf{s_t}|\mathbf{s_{t-1}^{[i]}},\mathbf{z_t}) = \frac{1}{K_i}\sum_{j=1}^{K_i} \delta_{\mathbf{s_t^{[i,j]}}}(\mathbf{s_t})$$

where $\{\mathbf{s_t^{[i,j]}}; \mathbf{j} = \mathbf{1},\ldots,\mathbf{K_i}\}$ represents the set of accepted samples corresponding to the i'th particle in time $t-1$. The target distribution is estimated by assigning weights to these samples according to (11). By applying total probability theorem, (11) can be rewritten as

$$w_{0:t}^{[i,j]} = w_{0:t-1}^{[i]} \int p(\mathbf{z_t}|\mathbf{s_t})\mathbf{p}(\mathbf{s_t}|\mathbf{s_{t-1}^{[i]}})\mathbf{ds_t}$$

Finally, using $p(\mathbf{s_t}|\mathbf{s_{t-1}^{[i]}}) \approx \frac{1}{M}\sum_{j=1}^{M} \delta_{\mathbf{s_t^{[i,j]}}}(\mathbf{s_t})$, importance weights can be obtained as

$$w_{0:t}^{[i,j]} \approx w_{0:t-1}^{[i]}[\frac{1}{M}\sum_{j=1}^{M} p(\mathbf{z_t}|\mathbf{s_t^{[i,j]}})] \qquad (17)$$

These weights are normalized later in order to obtain $\tilde{w}_{0:t}^{[i,j]}$.

Unlike FastSLAM, in order to avoid the error and depletion caused by unnecessary resampling steps, we only resample whenever N_{eff} gets below a threshold as it was discussed in section 3.

4.2 LMC-2

Another MC sampling method that can be used to generate samples from the OPD is importance sampling (IS). In this way, samples drawn from the local importance function $q(\mathbf{s_t}|\mathbf{s_{t-1}^{[i]}},\mathbf{z_t})$ receive local weights $w_t^{(\text{IS}),[i,j]} \propto \frac{p(\mathbf{s_t}|\mathbf{s_{t-1}^{[i]}},\mathbf{z_t})}{q(\mathbf{s_t}|\mathbf{s_{t-1}^{[i]}},\mathbf{z_t})}$ in order to approximate $p(\mathbf{s_t}|\mathbf{s_{t-1}^{[i]}},\mathbf{z_t})$ as

$$\hat{p}(\mathbf{s_t}|\mathbf{s_{t-1}^{[i]}},\mathbf{z_t}) = \sum_{j=1}^{M} \tilde{w}_t^{(\text{IS}),[i,j]} \delta_{\mathbf{s_t^{[i,j]}}}(\mathbf{s_t}) \qquad (18)$$

in which

$$\tilde{w}_t^{(\text{IS}),[i,j]} = \frac{w_t^{(\text{IS}),[i,j]}}{\sum_{j=1}^{M} w_t^{(\text{IS}),[i,j]}}$$

A straightforward choice is to select the SLAM motion model $p(\mathbf{s_t}|\mathbf{s_{t-1}^{[i]}})$ as the local importance function $q(\mathbf{s_t}|\mathbf{s_{t-1}^{[i]}},\mathbf{z_t})$ which implies

$$w_t^{(\text{IS}),[i,j]} = p(\mathbf{z_t}|\mathbf{s_t^{[i,j]}})$$

Similar to LMC-1, $p(\mathbf{z_t}|\mathbf{s_t^{[i,j]}})$ is evaluated as (16).

It can be easily shown that local weights $w_t^{(\text{IS}),[i,j]}$ must be multiplied by the SIS weights (11), in order to obtain the path posterior. We denote by $\pi^{\text{opt}}(\mathbf{s_{0:t}}|\mathbf{z_{1:t}})$ and

[4]An alternative approach would consist of generating T i.i.d. samples according to $p(\theta_{\mathbf{n_t}}|\mathbf{s_{0:t-1}^{[i]}},\mathbf{z_{0:t-1}})$ and then approximating $p(\mathbf{z_t}|\mathbf{s_t^{[i,j]}})$ as $\frac{1}{T}\sum_{l=1}^{T} p(\mathbf{z_t}|\mathbf{s_t^{[i,j]}},\theta_{\mathbf{n_t}}^{[\mathbf{l}]})$.

$\pi^{\text{prior}}(\mathbf{s_{0:t}}|\mathbf{z_{1:t}})$ respectively, the optimal and prior importance functions. Iterating (7) yields

$$\pi^{\text{opt}}(\mathbf{s_{0:t}}|\mathbf{z_{1:t}}) = p(\mathbf{s_0}) \prod_{\mathbf{i=1}}^{\mathbf{t}} \mathbf{p}(\mathbf{s_i}|\mathbf{s_{i-1}}, \mathbf{z_i})$$

$$\pi^{\text{prior}}(\mathbf{s_{0:t}}|\mathbf{z_{1:t}}) = p(\mathbf{s_0}) \prod_{\mathbf{i=1}}^{\mathbf{t}} \mathbf{p}(\mathbf{s_i}|\mathbf{s_{i-1}})$$

Finally, total weight is obtained as

$$\frac{p(\mathbf{s_{0:t}}|\mathbf{z_{0:t}})}{\pi^{\text{opt}}(\mathbf{s_{0:t}}|\mathbf{z_{1:t}})} \prod_{i=1}^{t} \frac{p(\mathbf{s_i}|\mathbf{s_{i-1}}, \mathbf{z_i})}{p(\mathbf{s_i}|\mathbf{s_{i-1}})} = \frac{p(\mathbf{s_{0:t}}|\mathbf{z_{0:t}})}{\pi^{\text{prior}}(\mathbf{s_{0:t}}|\mathbf{z_{1:t}})}$$

where according to (10) we have

$$w_{0:t}^{[i,j]} = w_{0:t-1}^{[i]} p(\mathbf{z_t}|\mathbf{s_t^{[i,j]}}) \propto \frac{\mathbf{p}(\mathbf{s_{0:t}^{[i,j]}}|\mathbf{z_{0:t}})}{\pi^{\text{prior}}(\mathbf{s_{0:t}^{[i,j]}}|\mathbf{z_{1:t}})}$$

Although LMC-2 is derived based on the idea of approximating the OPD, the result can also be interpreted as a modified FastSLAM 1.0 (i.e. prior proposal distribution) in which, instead of sampling one particle, M samples are generated according to $p(\mathbf{s_t}|\mathbf{s_{t-1}^{[i]}})$. It is of utmost importance to note that in this case, the most recent observation is playing part through $w_t^{(\text{IS}),[i,j]}$. In practice, we use $w_t^{(\text{IS}),[i,j]}$ in order to filter samples $\mathbf{s_t^{[i,j]}}$ with insignificant local weights. Moreover, we propagate only the samples with highest local weights whenever resampling is not performed. Therefore, for each i ($i = 1, \dots, N$) we have

$$\mathbf{s_t^{[i]}} = \arg\max_{\mathbf{s_t^{[i,j]}}} \mathbf{p}(\mathbf{z_t}|\mathbf{s_t^{[i,j]}})$$

whenever N_{eff} is above the threshold.

5 Results

5.1 Simulation Results

To evaluate the performance of the proposed SLAM algorithms and to systematically compare different methods of approximating the OPD, we use extensive simulations. We used the FastSLAM 2.0 implementation of Bailey *et al.* [Bailey *et al.*, 2006] and implemented our algorithms (LMC-1 and LMC-2) for the same environment. A Dual-Core 1.6 GHz PC with 2 GB RAM was used to perform the simulations. The vehicle motion and observation models and simulation parameters are similar to [Bailey *et al.*, 2006] and are not brought here for brevity. The simulated large scale environment ($\approx 200m \times 230m$), the robot trajectory and the estimated map are shown in Fig. 1. In each experiment, robot closes the loop once. In order to rule out the effect of data association errors

Table 1: Average number of resampling steps over 50 MC simulations

N	FastSLAM 2.0	LMC-1 ($M = 50$)	LMC-2 ($M = 3$)
20	221.01	180.84	188.85
40	235.71	186.55	195.33
60	239.55	189.62	195.99
80	243.61	190.39	197.96
100	245.70	192.69	199.50

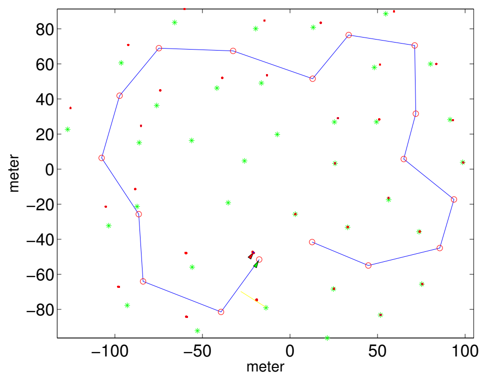


Figure 1: Simulated environment. The stars denote the landmark locations, the line represents the robot trajectory and red dots indicate the landmark estimates of particles.

on the performance of OPD approximation, true association is used in the simulations. For each experiment, 50 MC runs were performed. Therefore, mean squared error (MSE) is computed according to $\frac{1}{50}\sum_{b=1}^{50}(\mathbf{s_t^b} - \mathbf{\hat{s}_t^b})^2$ where $\mathbf{s_t^b}$ and $\hat{\mathbf{s}}_t^b$, respectively, denote the true and the estimated robot pose in time t of the b-th MC simulation. The estimated robot pose, $\mathbb{E}[\mathbf{s_t}|\mathbf{z_{1:t}}]$, is obtained by computing the weighted average of particles before resampling. It is of crucial importance to note that resampling will introduce additional "random variation" to the estimated robot pose and therefore, it is necessary to compute the desired expected value using the set of weighted particles before resampling [Liu and Chen, 1998]. The original FastSLAM 2.0 performs resampling whenever an observation is available. Here, an improved version of FastSLAM 2.0 is used in which resampling is performed based on the ESS (9). For all of the experiments, $N_{thr} = \frac{3}{4}N$ is used.

The main function of the OPD is to limit the degeneracy problem. It is crucial to note that in practice, by approximating the OPD we will only be able to reduce the rate of particle degeneration. Resampling addresses this issue partially. However, as it was mentioned earlier, resampling itself will lead to the other major issue in particle filtering, the so-called depletion problem in

Table 2: Average Runtime over 50 MC simulations

N	FastSLAM 2.0 (sec)	LMC-1 ($M = 50$) (sec)	LMC-2 ($M = 3$) (sec)
20	36	318	38
40	68	635	75
60	103	953	112
80	134	1265	148
100	167	1588	183

Table 3: Average MSE over the path

N	FastSLAM 2.0 (m^2)	LMC-1 ($M = 50$) (m^2)	LMC-2 ($M = 3$) (m^2)
10	65.5	53.52	48.77
40	44.75	43.63	43.99
60	46.87	41.29	41.55
100	40.44	33.54	43.46
200	44.21	24.62	41.82

which all the particles $\mathbf{s}_{\mathbf{0:t}}^{[\mathbf{i}]}$ at time step t share a common history of robot poses $\mathbf{s}_{\mathbf{0:t-k}}^{[\mathbf{i}]}$ and map for some k. Maintaining the diversity of particles in RBPF-SLAM is crucial to the efficient loop-closure performance. ESS measures the degeneracy in each time step and can be employed in order to determine if resampling is needed. Therefore, by using ESS, number of resamplings reflects both the rate of particle degeneration and depletion. For this reason, it is common to monitor and report the value of N_{eff} throughout the simulation. Nevertheless, due to the great deal of variation of N_{eff} among different MC trials for fixed time steps, we chose to report directly the average number of resampling steps among 50 MC runs.

Table 1 clearly shows that LMC-1 and LMC-2 outperform FastSLAM 2.0 in the number resampling steps performed in each experiment. LMC-1 is slightly better than LMC-2 in the number of resampling steps. Smaller number of resampling steps suggests slower rate of degeneracy and consequently, closer approximation of the OPD. As it was discussed above, this decrease in the number of resampling steps reduces the rate of loss of particle diversity and depletion. Reducing the rate of diversity loss significantly affects the SLAM algorithm performance as it is suggested by [Bailey *et al.*, 2006].

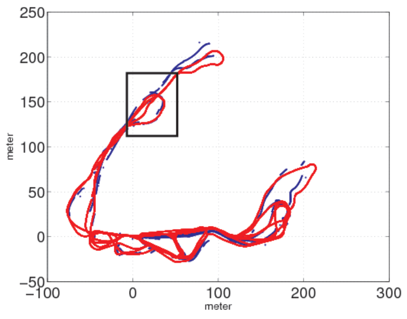
The average runtime of LMC-1, LMC-2 and FastSLAM 2.0 in the aforementioned scenario is reported in Table 2. FastSLAM 2.0 is slightly faster than LMC-2 ($M = 3$) while LMC-1 ($M = 50$) requires approximately 10 times more computational time to process a loop. It is important to note that LMC-1 uses local samples to approximate the integral in (17) in order to calculate the optimal weights. This MC integration requires a fairly large number of local samples, M, in order to obtain accurate results. On the contrary, in LMC-2, optimal weights are eliminated thus good estimates can be obtained using much smaller M.

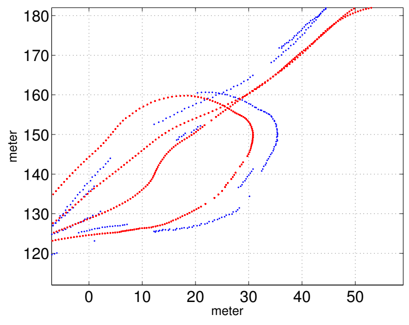
The performance of these algorithms are also compared by varying the number of particles and computing the average MSE over time. The result is averaged over 50 MC trials and is reported in Table 3. According to Table 3, LMC-2 ($M = 3$) exhibits the lowest MSE for small number of particles ($N = 10$), followed closely by LMC-1 ($M = 50$). On the contrary, FastSLAM 2.0 shows quite a large MSE compared to LMC-2 and LMC-1 for 10 particles. The accuracy of FastSLAM 2.0 and LMC-2 ($M = 3$) becomes more similar as the number of particles increases. This behavior should not come as a surprise, because it is well-known that for "large enough" number of particles the difference between sampling from different proposal distributions diminishes. Nevertheless, LMC-1 ($M = 50$) yields even more accurate estimates as the number of particles increases. Therefore, LMC-1 ($M = 50$) outperforms the other two algorithms in terms of accuracy for large number of particles, while LMC-2 ($M = 3$) has shown the lowest MSE for small number of particles.

Finally, the MSE of estimated robot position throughout the loop for different algorithms with fixed number of particles ($N = 80$) is shown in Fig. 2. FastSLAM 2.0 and LMC-1 ($M = 20$) have failed to obtain good estimates. The poor performance of LMC-1 ($M = 20$) suggests that $M = 20$ is not enough to approximate the integral in optimal weights (17). Moreover, the fact that LMC-2 is able to achieve much more accurate estimates with the same and lower ($M = 3$) number of local samples supports this claim. According to Fig. 2, LMC-1 ($M = 50$) has the lowest MSE at the expense of its huge computational time. Note that FastSLAM 2.0 has shown the largest MSE at the end of path. This behavior is consistent with the FastSLAM 2.0's larger number of resamplings which adversely affects its loop-closing performance.

5.2 Victoria Park Dataset Results

Due to the stochastic nature of SMC methods, MC simulations are necessary for evaluation of the proposed algorithms. However, in order to evaluate the performance of the proposed algorithms in real-world situations, they were also tested on the well-known Victoria Park dataset from the University of Sydney. The dataset was gathered by using a vehicle equipped with a SICK laser range finder (LRF), driven in a large-scale environment ($\approx 250m \times 250m$). The length of the traversed path is approximately 3.5 km. The trees in the park were extracted from the LRF scans and are used as the point features in feature-based SLAM algorithms. Neverthe-


(a) LMC-1 ($M = 30$)


(b) LMC-2 ($M = 3$)

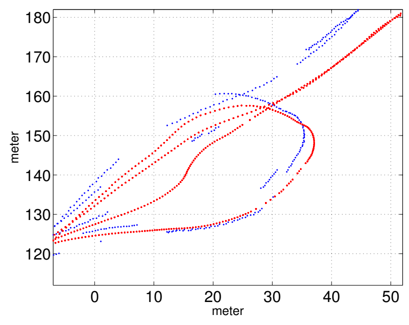

(c) LMC-1 ($M = 30$)

Figure 3: The estimated trajectory (red) and GPS data (blue) for 20 particles. The zoomed version of the boxed region in Fig. 3a for each algorithm is shown in Fig. 3b and 3c.

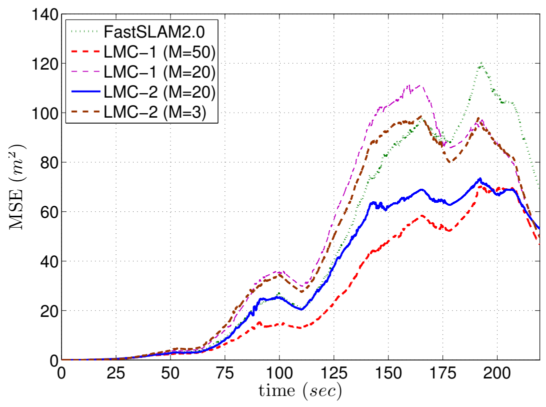


Figure 2: Average MSE of estimated robot position over 50 Monte Carlo simulations.

less, the dataset is notorious for spurious observations which makes the data association difficult. GPS data is available partially and is used as the ground truth, but there is no ground truth available for the landmarks positions.

Similar to the simulations, data association is assumed known here. We used the set of associations provided by I-SLSJF algorithm [Huang and Wang, 2008]. Since there is no ground truth available for the landmarks positions, only the estimated robot trajectories are plotted here. Number of resamplings for each algorithm with 20 particles is shown in Table 4. According to Table 4, LMC-1 ($M = 30$) has the lowest number of resampling steps, followed by LMC-2 ($M = 3$). Compared to FastSLAM 2.0, LMC-1 and LMC-2 have reduced the number of resamplings by 27% and 20% respectively. These results are consistent with those obtained from the simulations.

The accuracy of LMC-1 ($M = 30$), LMC-2 ($M = 3$) and FastSLAM 2.0 with 20 particles were quite similar in the first part of the trajectory. The estimated trajectory

Table 4: Number of Resamplings in Victoria Park Dataset

N	FastSLAM 2.0	LMC-1 ($M = 30$)	LMC-2 ($M = 3$)
20	1900	1394	1528

for LMC-1 is shown in Fig. 3a. The zoomed version of the boxed region in Fig. 3a for each algorithm is depicted in Fig. 3b and 3c. According to Fig. 3 LMC-1 ($M = 30$) has shown the best performance, followed by LMC-2 ($M = 3$). Such results were expected according to Table 4 and simulations results. The error can be reduced further by increasing M and/or N.

6 Conclusion

In this paper we reviewed different approaches to approximate the OPD in the feature-based RBPF-SLAM. Furthermore, two novel feature-based SLAM algorithms based on MC approximation of the OPD have been proposed. The proposed algorithms employ MC sampling methods to draw samples from the non-Gaussian OPD of SLAM by considering it as a local target. Both of these algorithms benefit from the capability of MC sampling methods to represent and sample from non-Gaussian and multi-modal distributions, and consequently outperform Gaussian approximation approaches such as FastSLAM 2.0. To the best of our knowledge, the proposed algorithms are the first solutions of the feature-based SLAM with such a capability. Finally, the proposed algorithms were evaluated using both simulations and real-world data in large scale environments. Results have shown improvements compared to FastSLAM 2.0. The accuracy of LMC-1 comes at the expense of huge computational time which makes it ideal for offline applications. On the other hand, while LMC-2 is as fast as FastSLAM 2.0, it yields much more accurate estimates than Fast-

SLAM (especially for small number of particles).

References

[Bailey *et al.*, 2006] T. Bailey, J. Nieto, and E. Nebot. Consistency of the fastslam algorithm. In *Proc. IEEE Int. Conf. Robotics and Automation ICRA 2006*, pages 424–429, 2006.

[Blanco *et al.*, 2010] J. Blanco, J. Gonzalez, and J.A. Fernandez-Madrigal. Optimal Filtering for Non-parametric Observation Models: Applications to Localization and SLAM. *The International Journal of Robotics Research*, 2010.

[Caffo *et al.*, 2002] B.S.Caffo, J.G. Booth, and AC Davison. Empirical supremum rejection sampling. *Biometrika*, 89(4):745, 2002.

[Douc and Cappé, 2005] R. Douc and O. Cappé. Comparison of resampling schemes for particle filtering. In *Image and Signal Processing and Analysis, 2005. ISPA 2005. Proceedings of the 4th International Symposium on*, pages 64–69. IEEE, 2005.

[Doucet *et al.*, 2001] A. Doucet, N. De Freitas, and N. Gordon. *Sequential Monte Carlo methods in practice.* Springer Verlag, 2001.

[Doucet, 1998] Arnaud Doucet. On sequential simulation-based methods for bayesian filtering. Technical Report CUED/F-INFENG/TR. 310, Cambridge University Department of Engineering, 1998.

[Gordon *et al.*, 1993] N.J. Gordon, D.J. Salmond, and A.F.M. Smith. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. In *IEE Proceedings*, volume 140, pages 107–113, 1993.

[Grisetti *et al.*, 2007] G. Grisetti, C. Stachniss, and W. Burgard. Improved techniques for grid mapping with rao-blackwellized particle filters. *Robotics, IEEE Transactions on*, 23(1):34–46, 2007.

[Huang and Wang, 2008] S. Huang and Z Wang. I-slsjf implementation, 2008. http://www.openslam.org/2d-i-slsjf.html

[Kim *et al.*, 2008] C. Kim, R. Sakthivel, and W.K. Chung. Unscented fastslam: A robust and efficient solution to the slam problem. *Robotics, IEEE Transactions on*, 24(4):808–820, 2008.

[Kong *et al.*, 1994] A. Kong, J.S. Liu, and W.H. Wong. Sequential imputations and Bayesian missing data problems. *Journal of the American Statistical Association*, 89(425):278–288, 1994.

[Liu and Chen, 1998] Jun S. Liu and Rong Chen. Sequential monte carlo methods for dynamic systems. *Journal of the American Statistical Association*, 93:1032–1044, 1998.

[Montemerlo *et al.*, 2002] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM: A factored solution to the simultaneous localization and mapping problem. In *Proceedings of the National conference on Artificial Intelligence*, pages 593–598. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2002.

[Montemerlo *et al.*, 2003] 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 Sixteenth International Joint Conference on Artificial Intelligence (IJCAI)*, Acapulco, Mexico, 2003. IJCAI.

[Murphy, 2000] K. Murphy. Bayesian map learning in dynamic environments. *Advances in Neural Information Processing Systems*, 12:1015–1021, 2000.

[Neumann, 1951] J. Neumann. Various techniques used in connection with random digits. Monte Carlo methods. *National Bureau of Standards AMS*, 12, 1951.

[Saha *et al.*, 2009] S. Saha, PK Mandal, Y. Boers, H. Driessen, and A. Bagchi. Gaussian proposal density using moment matching in smc methods. *Statistics and Computing*, 19(2):203–208, 2009.

[Stachniss *et al.*, 2007] C.Stachniss, G.Grisetti, W. Burgard, and N. Roy. Analyzing gaussian proposal distributions for mapping with rao-blackwellized particle filters. In *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pages 3485–3490. IEEE, 2007.

[Van Der Merwe *et al.*, 2001] R.VanDer Merwe, A. Doucet, N. De Freitas, and E. Wan. The unscented particle filter. *Advances in Neural Information Processing Systems*, pages 584–590, 2001.