FastSLAM 2.0: An Improved Particle Filtering Algorithm for Simultaneous Localization and Mapping that Provably Converges

Michael Montemerlo and **Sebastian Thrun**
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
mmde@cs.cmu.edu, thrun@cs.cmu.edu

Daphne Koller and **Ben Wegbreit**
Computer Science Department
Stanford University
Stanford, CA 94305-9010
koller@cs.stanford.edu, ben@wegbreit.com

Abstract

In [15], Montemerlo et al. proposed an algorithm called *FastSLAM* as an efficient and robust solution to the simultaneous localization and mapping problem. This paper describes a modified version of FastSLAM that overcomes important deficiencies of the original algorithm. We prove convergence of this new algorithm for linear SLAM problems and provide real-world experimental results that illustrate an order of magnitude improvement in accuracy over the original FastSLAM algorithm.

1 Introduction

Simultaneous localization and mapping (SLAM) is a highly active research area in robotics and AI. The SLAM problem arises when a moving vehicle (e.g. a mobile robot, submarine, or drone) simultaneously estimates a map of its environment and its pose relative to that map. In the absence of global position information, the vehicle's pose estimate will become increasingly inaccurate, as will its map. Since maps may contain thousands of entities, acquiring large, accurate maps is a challenging statistical estimation problem, especially when performed in real-time.

Most present-day research on SLAM originates from a seminal paper by Smith and Cheeseman [21], which proposed the use of the extended Kalman filter (EKF) for solving SLAM. This paper is based on the insights that errors in the map and pose errors are naturally correlated, and that the covariance matrix maintained by the EKF expresses such correlations. Newmann [18] recently proved that the EKF converges for linear SLAM problems, where the motion model and observation model are linear functions with Gaussian noise (see below).

Unfortunately, EKF covariance matrices are quadratic in the size of the map, and updating them requires time quadratic in the number of landmarks N. This quadratic complexity has long been recognized to be a major obstacle in scaling SLAM algorithms to maps with more than a few hundred features. It also limits the applicability of SLAM algorithms to problems with ambiguous landmarks, which induces a *data association problem* [2; 22]. Today's most robust algorithms for SLAM with unknown data association maintain multiple hypotheses (tracks), which increase their computational complexity.

Consequently, there has been a flurry on research on more efficient SLAM techniques (see e.g., [11]). One group of researchers has developed techniques that recursively divide maps into submaps, thereby confining most computation to small regions. Some of these approaches still maintain global correlations among those submaps, hence are quadratic but with a much reduced constant factor [1; 7; 22; 25]. Others restrict the update exclusively to local maps [12], hence operate in constant time (assuming known data association).

A second group of researchers has developed techniques that represent maps through potential functions between adjacent landmarks, similar to Markov random fields. The resulting representations require memory linear in the number of landmarks [19; 23]. Under appropriate approximations, such techniques have been shown to provide constant time updating (again for known data association). Unfortunately, no convergence proof exists for any of these extensions of the EKF, even for the generic case of linear SLAM. Furthermore, if landmarks are ambiguous, all of these approaches have to perform search to find appropriate data association hypotheses, adding a logarithmic factor to their update complexity.

The FastSLAM algorithm, proposed in [15] as an efficient approach to SLAM based on particle filtering [6], does not fall into either of the categories above. FastSLAM takes advantage of an important characteristic of the SLAM problem (with known data association): landmark estimates are conditionally independent given the robot's path [17]. FastSLAM uses a particle filter to sample over robot paths. Each particle possesses N low-dimensional EKFs, one for each of the N landmarks. This representation requires $O(NM)$ memory, where M is the number of particles in the particle filter. Updating this filter requires $O(M \log N)$ time, with or without knowledge of the data associations. However, the number of particles M needed for convergence is unknown and has been suspected to be exponential in the size of the map, in the worst-case.

This paper proposes an improved version of the FastSLAM algorithm. The modification is conceptually simple: When proposing a new robot pose—an essential step in FastSLAM's particle filter—our proposal distribution relies not only on the motion estimate (as is the case in FastSLAM), but also on the most recent sensor measurement. Such an approach is less wasteful with its samples than the original FastSLAM algorithm, especially in situations where the noise in motion is high relative to the measurement noise.

To obtain a suitable proposal distribution, our algorithm linearizes the motion and the measurement model in the same manner as the EKF. As a result, the proposal distribution can be calculated in closed form. This extension parallels prior work by Doucet and colleagues, who proposed a similar modification for general particle filters [6] and Markov Chain Monte Carlo techniques for neural networks [4]. It is similar to the arc reversal technique proposed for particle filters applied to

Bayes networks [10], and it is similar to recent work by van der Merwe [24], who uses an unscented filtering step [9] for generating proposal distributions that accommodate the measurement.

While this modification is conceptually simple, it has important ramifications. A key contribution of this paper is a convergence proof for linear SLAM problems using a single particle. The resulting algorithm requires constant updating time. To our knowledge, the best previous SLAM algorithm for which convergence was shown requires quadratic update time. Furthermore, we observe experimentally that our new FastSLAM algorithm, even with a single particle, yields significantly more accurate results on a challenging real-world benchmark [7] than the previous version of the algorithm. These findings are of significance, as many mobile robot systems are plagued by control noise, but possess relatively accurate sensors. Moreover, they contradict a common belief that maintaining the entire covariance matrix is required for convergence [5].

2 Simultaneous Localization and Mapping

SLAM addresses the problem of simultaneously recovering a map and a vehicle pose from sensor data. The map contains N features (landmarks) and shall be denoted $\Theta = \theta_1, \ldots, \theta_N$. The path of the vehicle will be denoted $s^t = s_1, \ldots, s_t$, where t is a time index and s_t is the pose of the vehicle at time t.

Most state-of-the-art SLAM algorithms calculate (or approximate) variants of the following posterior distribution:

$$p(\Theta, s^t \mid z^t, u^t, n^t) \tag{1}$$

where $z^t = z_1, \ldots, z_t$ is a sequence of measurements (e.g., range and bearing to nearby landmarks), and $u^t = u_1, \ldots, u_t$ is a sequence of robot controls (e.g., velocities for robot wheels). (As usual, we assume without loss of generality that only a single landmark is observed at each time t.) The variables $n^t = n_1, \ldots, n_t$ are *data association variables* — each n_t specifies the identity of the landmark observed at time t. Initially, we assume n^t is known; we relax this assumption below.

To calculate the posterior (1), the vehicle is given a probabilistic motion model, in the form of the conditional probability distribution $p(s_t \mid u_t, s_{t-1})$. This distribution describes how a control u_t, asserted in the time interval $[t-1; t)$, affects the resulting pose. Additionally, the vehicle is given a probabilistic measurement model, denoted $p(z_t \mid s_t, \Theta, n_t)$, describing how measurements evolve from state. In accordance to the rich SLAM literature, we will model both models by nonlinear functions with independent Gaussian noise:

$$p(z_t \mid s_t, \Theta, n_t) = g(s_t, \theta_{n_t}) + \varepsilon_t \tag{2}$$

$$p(s_t \mid u_t, s_{t-1}) = h(u_t, s_{t-1}) + \delta_t \tag{3}$$

Here g and h are nonlinear functions, and ε_t and δ_t are Gaussian noise variables with covariance R_t and P_t, respectively.

3 FastSLAM

FastSLAM [15] is based on the important observation [17] that the posterior can be factored

$$p(\Theta, s^t \mid z^t, u^t, n^t) = p(s^t \mid z^t, u^t, n^t) \prod_n p(\theta_n \mid s^t, z^t, u^t, n^t) \tag{4}$$

This factorization is exact and universal in SLAM problems. It states that if one (hypothetically) knew the path of the vehicle, the landmark positions could be estimated independently of each other (hence the product over n). In practice, of course, one does not know the vehicle's path. Nevertheless, the independence makes it possible to factor the posterior into a term that estimates the probability of each path, and N terms that estimate the position of the landmarks, conditioned on each (hypothetical) path.

FastSLAM samples the path using a particle filter. Each particle has attached its own map, consisting of N extended Kalman filters. Formally, the m-th particle $S_t^{[m]}$ contains a path $s^{t,[m]}$ along with N Gaussian landmark estimates, described by the mean $\mu_{n,t}^{[m]}$ and covariance $\Sigma_{n,t}^{[m]}$:

$$S_t^{[m]} = s^{t,[m]}, \underbrace{\mu_{1,t}^{[m]}, \Sigma_{1,t}^{[m]}}_{\text{landmark } \theta_1}, \ldots, \underbrace{\mu_{N,t}^{[m]}, \Sigma_{N,t}^{[m]}}_{\text{landmark } \theta_N} \tag{5}$$

We briefly review the key equations of the regular FastSLAM algorithm, and refer the reader to [15]. Each update in FastSLAM begins with sampling new poses based on the most recent motion command u_t:

$$s_t^{[m]} \sim p(s_t \mid s_{t-1}^{[m]}, u_t). \tag{6}$$

Note that this proposal distribution only uses the motion command u_t, but *ignores* the measurement z_t.

Next, FastSLAM updates the estimate of the observed landmark(s), according to the following posterior. This posterior takes the measurement z_t into consideration:

$$p(\theta_{n_t} \mid s^{t,[m]}, n^t, z^t) \tag{7}$$
$$= \eta \underbrace{p(z_t \mid \theta_{n_t}, s_t^{[m]}, n_t)}_{\sim \mathcal{N}(z_t; g(\theta_{n_t}, s_t^{[m]}), R_t)} \underbrace{p(\theta_{n_t} \mid s^{t-1,[m]}, z^{t-1}, n^{t-1})}_{\sim \mathcal{N}(\theta_{n_t}; \mu_{n_t,t-1}^{[m]}, \Sigma_{n_t,t-1}^{[m]})}$$

Here η is a constant. This posterior is the normalized product of two Gaussians as indicated. However, if g is non-linear, the product will not be Gaussian in general. To make the result Gaussian, FastSLAM employs the standard EKF "trick" [13]: g is approximated by a linear function (see below). Under this approximation, (7) is equivalent to the measurement update equation familiar from the EKF literature [13].

In a final step, FastSLAM corrects for the fact that the pose sample $s_t^{[m]}$ has been generated without consideration of the most recent measurement. It does so by resampling the particles [20]. The probability for the m-th particle to be sampled (with replacement) is given by the following variable $w_t^{[m]}$, commonly referred to as *importance factor*:

$$w_t^{[m]} = \eta \int \underbrace{p(z_t \mid \theta_{n_t}, s_t^{[m]}, n_t)}_{\sim \mathcal{N}(z_t; g(\theta_{n_t}, s_t^{[m]}), R_t)} \underbrace{p(\theta_{n_t} \mid s^{t-1,[m]}, z^{t-1}, n^{t-1})}_{\sim \mathcal{N}(\mu_{n_t,t-1}^{[m]}, \Sigma_{n_t,t-1}^{[m]})} d\theta_{n_t}$$

As shown in [15], the resampling operation can be implemented in $O(M \log N)$ time using trees, where M is the number of samples and N the number of landmarks in the map. However, the number of particles M needed for convergence remains an open question.

FastSLAM has been extended to SLAM with unknown data associations [14]. If the data association is unknown, each particle m in FastSLAM makes its own local data association decision $\hat{n}_t^{[m]}$, by maximizing the measurement likelihood. The formula for finding the most likely data association maximizes the resulting importance weight:

$$\hat{n}_t^{[m]} = \operatorname*{argmax}_{n_t} w_t^{[m]}(n_t) \tag{8}$$

Here $w_t^{[m]}(n_t)$ makes the dependence of the factor $w_t^{[m]}$ on the variable n_t explicit. A key characteristic of FastSLAM is that each particle makes its own local data association. In contrast, EKF techniques must commit to a single data association hypothesis for the entire filter. Results in [14] show empirically that this difference renders FastSLAM significantly more robust to noise than EKF-style algorithms.

4 FastSLAM 2.0

Our new FastSLAM algorithm is based on an obvious inefficiency arising from the proposal distribution of regular FastSLAM. In regular FastSLAM, the pose $s_t^{[m]}$ is sampled in accordance to the prediction arising from the motion command u_t, as specified in (6). It does not consider the measurement z_t acquired at time t; instead, the measurement is incorporated through resampling. This approach is particularly troublesome if the noise in the vehicle motion is large relative to the measurement noise. In such situations, sampled poses will mostly fall into areas of low measurement likelihood, and will subsequently be terminated in the resampling phase with high probability. Unfortunately, many real-world robot systems are characterized by relatively high motion noise. As illustrated in the experimental results section of this paper, the waste incurred by this inefficient sampling scheme can be significant.

4.1 Sampling The Pose

FastSLAM 2.0 implements a single new idea: Poses are sampled under consideration of both the motion u_t and the measurement z_t. This is formally denoted by the following sampling distribution, which now takes the measurement z_t into consideration:

$$s_t^{[m]} \quad \sim \quad p(s_t \mid s^{t-1,[m]}, u^t, z^t, n^t) \tag{9}$$

In comparison to (6), incorporating the measurement only makes sense if we incorporate our current estimate of the observed landmark—obtained from the variables $s^{t-1,[m]}, u^{t-1}, z^{t-1}, n^{t-1}$ (which are included of the conditioning variables above). So in essence, the difference to FastSLAM is that the measurement z_t is incorporated. However, this change has important ramifications.

The proposal distribution (9) can be reformulated as follows:

$$\begin{aligned} & p(s_t \mid s^{t-1,[m]}, u^t, z^t, n^t) \\ = \quad & \eta^{[m]} \int \underbrace{p(z_t \mid \theta_{n_t}, s_t, n_t)}_{\sim \mathcal{N}(z_t; g(\theta_{n_t}, s_t), R_t)} \underbrace{p(\theta_{n_t} \mid s^{t-1,[m]}, z^{t-1}, n^{t-1})}_{\sim \mathcal{N}(\theta_{n_t}; \mu^{[m]}_{n_t,t-1}, \Sigma^{[m]}_{n_t,t-1})} \, d\theta_{n_t} \\ & \underbrace{p(s_t \mid s_{t-1}^{[m]}, u_t)}_{\sim \mathcal{N}(s_t; h(s^{[m]}_{t-1}, u_t), P_t)} \end{aligned} \tag{10}$$

That is, the proposal distribution is the product of two factors: the familiar next state distribution $p(s_t \mid s_{t-1}^{[m]}, u_t)$, and the probability of the measurement z_t. Calculating the latter involves an integration over possible landmark locations θ_{n_t}.

Unfortunately, sampling directly from this distribution is impossible in the general case; it does not even possess a closed form. Luckily, a closed form solution can be attained if g is approximated by a linear function (h may remain non-linear!):

$$g(\theta_{n_t}, s_t) \quad \approx \quad \hat{z}_t^{[m]} + G_\theta \cdot (\theta_{n_t} - \mu^{[m]}_{n_t,t-1}) + G_s \cdot (s_t - \hat{s}_t^{[m]})$$

where $\hat{z}_t^{[m]} = g(\hat{\theta}^{[m]}_{n_t}, \hat{s}_t^{[m]})$ denotes the predicted measurement, $\hat{s}_t^{[m]} = h(s_{t-1}^{[m]}, u_t)$ the predicted robot pose, and $\hat{\theta}_n^{[m]} = \mu_{n,t-1}^{[m]}$ the predicted landmark location. The matrices G_θ and G_s are the Jacobians (first derivatives) of g with respect to θ and s, respectively:

$$G_\theta \quad = \quad \nabla_{\theta_{n_t}} g(\theta_{n_t}, s_t)\big|_{s_t = \hat{s}_t^{[m]}; \theta_{n_t} = \hat{\theta}^{[m]}_{n_t}} \tag{11}$$

$$G_s \quad = \quad \nabla_{s_t} g(\theta_{n_t}, s_t)\big|_{s_t = \hat{s}_t^{[m]}; \theta_{n_t} = \hat{\theta}^{[m]}_{n_t}} \tag{12}$$

Under this EKF-style approximation, the proposal distribution (9) is Gaussian with the following parameters:

$$\Sigma_{s_t}^{[m]} \quad = \quad \left[G_s^T Q_t^{[m]-1} G_s + P_t^{-1} \right]^{-1} \tag{13}$$

$$\mu_{s_t}^{[m]} \quad = \quad \Sigma_{s_t}^{[m]} G_s^T Q_t^{[m]-1} (z_t - \hat{z}_t^{[m]}) + \hat{s}_t^{[m]} \tag{14}$$

where the matrix $Q_t^{[m]}$ is defined as follows:

$$Q_t^{[m]} \; = \; R_t + G_\theta \Sigma^{[m]}_{n_t,t-1} G_\theta^T \tag{15}$$

4.2 Updating The Observed Landmark Estimate

The updating step remains the same as in FastSLAM (see (7)). As stated in the previous section, g is linearized to retain Gaussianity of the posterior. This leads to the following update equations, whose derivation is equivalent to that of the standard EKF measurement update [13]:

$$K_t^{[m]} \quad = \quad \Sigma^{[m]}_{n_t,t-1} G_\theta^T Q_t^{[m]-1} \tag{16}$$

$$\mu^{[m]}_{n_t,t} \quad = \quad \mu^{[m]}_{n_t,t-1} + K_t^{[m]} (z_t - \hat{z}_t^{[m]}) \tag{17}$$

$$\Sigma^{[m]}_{n_t,t} \quad = \quad (I - K_t^{[m]} G_\theta) \Sigma^{[m]}_{n_t,t-1} \tag{18}$$

4.3 The Importance Weights

Resampling is necessary even in our new version of FastSLAM, since the particles generated do not yet match the desired posterior. The culprit is the normalizer $\eta^{[m]}$ in (10), which may be different for different particles m. This normalizer is the inverse of the probability of the measurement under the m-th particle: $\eta^{[m]} = p(z_t \mid s^{t-1,[m]}, u^t, z^{t-1}, n^t)^{-1}$. To account for this mismatch, our algorithm resamples in proportion to the following importance factor:

$$\begin{aligned} w_t^{[m]} \propto \quad & p(z_t \mid s^{t-1,[m]}, u^t, z^{t-1}, n^t) \\ = \quad & \int\int \underbrace{p(z_t \mid \theta_{n_t}, s_t, n_t)}_{\sim \mathcal{N}(z_t; g(\theta_{n_t}, s_t), R_t)} \\ & \underbrace{p(\theta_{n_t} \mid s^{t-1,[m]}, u^{t-1}, z^{t-1}, n^{t-1})}_{\sim \mathcal{N}(\theta_{n_t}; \mu^{[m]}_{n_t,t-1}, \Sigma^{[m]}_{n_t,t-1})} \, d\theta_{n_t} \; \underbrace{p(s_t \mid s_{t-1}^{[m]}, u_t)}_{\sim \mathcal{N}(s_t; \hat{s}^{[m]}_{t-1}, P_t)} \, ds_t \end{aligned}$$

This expression can once again be approximated as a Gaussian by linearizing g. The mean of this Gaussian is $\hat{z}_t$, and its covariance is

$$G_s P_t G_s^T + G_\theta \Sigma^{[m]}_{n_t,t-1} G_\theta^T + R_t \tag{19}$$

4.4 Unknown Data Associations

The approach for handling data association is similar to the one in regular FastSLAM: Again, we select the data association n_t that maximizes the probability of the measurement z_t for the m-th particle:

$$\hat{n}_t^{[m]} \quad = \quad \operatorname*{argmax}_{n_t} p(z_t \mid n_t, \hat{n}^{t-1,[m]}, s^{t,[m]}, z^{t-1}, u^t) \tag{20}$$

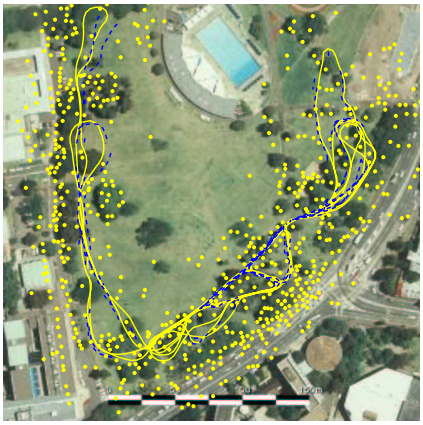
(a)  (b)  (c) 

Figure 1: FastSLAM 2.0 applied to the Victoria Park benchmark data set. (a) Raw vehicle odometry (b) FastSLAM 2.0, $M=1$ particle (c) FastSLAM 2.0 with dynamic feature management.

At first glance, one may be tempted to substitute $w_t^{[m]}$ for the probability on the right-hand side, as in regular FastSLAM. However, $w_t^{[m]}$ does not consider the sampled pose $s_t^{[m]}$, whereas the expression here does. This leads to a slightly different probability, which is calculated as follows.

$$p(z_t \mid n_t, \hat{n}^{t-1,[m]}, s^{t,[m]}, z^{t-1}, u^t) \tag{21}$$
$$= \int \underbrace{p(z_t \mid \theta_{n_t}, n_t, s_t^{[m]})}_{\sim\, \mathcal{N}(z_t; g(\theta_{n_t}, s_t^{[m]}), R_t)} \underbrace{p(\theta_{n_t} \mid \hat{n}^{t-1,[m]}, s^{t-1,[m]}, z^{t-1})}_{\sim\, \mathcal{N}(\mu_{n_t,t-1}^{[m]}, \Sigma_{n_t,t-1}^{[m]})} d\theta_{n_t}$$

Linearization of g leads to a Gaussian over z_t with mean $g(\mu_{n_t,t-1}^{[m]}, s_t^{[m]})$ and covariance $Q_t^{[m]}$. Both are functions of the data association variable n_t.

4.5 Feature Management

Finally, in cases with unknown data associations, features have to be created dynamically. As is common for SLAM algorithms [5], our approach creates new features when the measurement probability in (20) is below a threshold. However, real-world data with frequent outliers will generate spurious landmarks using this rule. Following [5], our approach removes such spurious landmarks by keeping track of their posterior probability of existence. Our mechanism analyzes measurement to the presence *and* absence of features. Observing a landmark provides positive evidence for its existence, whereas not observing it when $\mu_n^{[m]}$ falls within the robot's perceptual range provides negative evidence. The posterior probability of landmark existence is accumulated by the following Bayes filter, whose log-odds form is familiar from the literature on occupancy grid maps [16]:

$$\tau_n^{[m]} = \sum_t \ln \frac{p(i_n^{[m]} \mid s_t^{[m]}, z_t, \hat{n}_t^{[m]})}{1 - p(i_n^{[m]} \mid s_t^{[m]}, z_t, \hat{n}_t^{[m]})} \tag{22}$$

Here $\tau_n^{[m]}$ are the log-odds of the physical existence of landmark $\theta_n^{[m]}$ in map m, and $p(i_n^{[m]} \mid s_t^{[m]}, z_t, \hat{n}_t^{[m]})$ is the probabilistic evidence provided by a measurement. Under appropriate definition of the latter, this rule provides for a simple evidence counting rule. If the log odds drops below a predefined threshold, the corresponding landmark is removed from the map. This mechanism enables particles to free themselves of spurious features.

5 Convergence

A key result in this paper is the fact that our new version of FastSLAM converges for $M=1$ particle, for a restricted class of linear Gaussian problems (the same for which KFs converge [5; 18]). Specifically, our result applies to SLAM problems characterized by the following linear form:

$$g(s_t, \theta_{n_t}) = \theta_{n_t} - s_t \tag{23}$$
$$h(u_t, s_{t-1}) = u_t + s_{t-1} \tag{24}$$

Linear SLAM can be thought of as a robot operating in a Cartesian space equipped with a noise-free compass, and sensors that measure distances to features along the coordinate axes. The following theorem, whose proof can be found in the appendix, states the convergence of our new FastSLAM variant:

Theorem. *For linear SLAM, FastSLAM with $M=1$ particles converges in expectation to the correct map if all features are observed infinitely often, and if the location of one feature is known in advance.*

This theorem parallels a similar result previously published for the Kalman filter [5; 18]. However, this result applies to the Kalman filter, whose update requires time quadratic in the number of landmarks N. With $M=1$, the resampling step becomes obsolete and each update takes constant time. To our knowledge, our result is the first convergence result for a constant-time SLAM algorithm. It even holds if all features are arranged in a large loop, a situation often thought of as the worst case for SLAM problems [8].

6 Experimental Results

Systematic experiments showed that FastSLAM 2.0 provides excellent results with surprisingly few particles, including $M=1$. Most of our experiments were carried out using a benchmark data set collected with an outdoor vehicle in Victoria Park, Sydney [7]. The vehicle path is 3.5km long, and the map is 320 meters wide. The vehicle is equipped with differential GPS that is used for evaluation only. Fig. 1a shows the map of the terrain, along with the path obtained by raw odometry (which is very poor, the average RMS error is 93.6 meters). This data set is presently the most popular benchmark in the SLAM research community [3].

Figs. 1b&c show the result of applying FastSLAM with $M=1$ particle to the data set, without (Fig. 1b) and with (Fig. 1c) the feature management approach described in Section 4.5. In both cases, the estimated vehicle path is shown

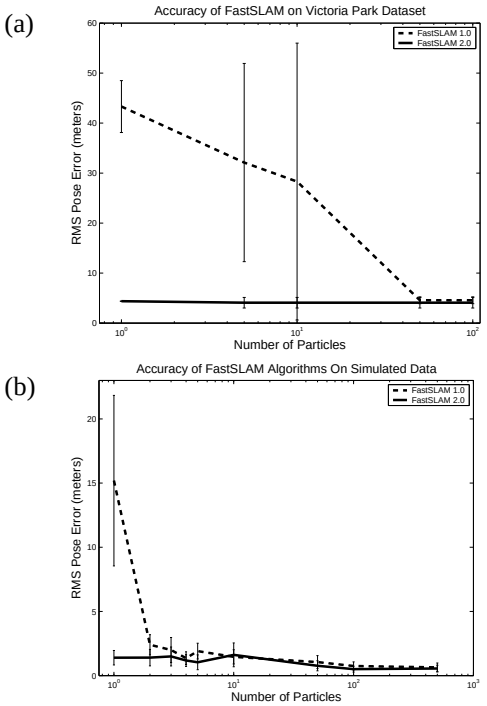


Figure 2: RMS map error for regular FastSLAM (dashed line) versus FastSLAM 2.0 (solid line) on (a) the Victoria Park data (b) simulated data. FastSLAM 2.0's results even with a single particle are excellent.

as a solid line, and the GPS information is shown as a dashed line. Results of the same accuracy were previously achieved only with $O(N^2)$ EKF-style methods [7] and with FastSLAM using $M{=}50$ particles. The feature management rule reduces the number of landmarks in the map from 768 (Fig. 1b) to 343 (Fig. 1c).

Fig. 2 plots the RMS error of the vehicle position estimate as function of the number of particles for the Victoria data set (panel a) and for synthetic simulation data (panel b) taken from [15]. While our new algorithm does approximately equally well for any number of particles, regular FastSLAM performs poorly for very small particle sets. We suspect that the poor performance of regular FastSLAM is due to the fact that the vehicle possesses relatively inaccurate odometry (see Fig. 1a), yet uses a low-noise range finder for landmark detection (a common configuration in outdoor robotics), leading to the generation of many particles of low likelihood.

The small number of examples needed to obtain state-of-the-art estimation translates to unprecedented efficiency of the new filter. The following table shows the results required to process the Victoria Park data set on a 1GHz Pentium PC:

EKF	7,807 sec
regular FastSLAM, $M{=}50$ particles	315 sec
FastSLAM 2.0, $M{=}1$ particle	54 sec

In comparison, the data acquisition required 1,550 seconds. Thus, while EKFs cannot be run in real-time, our new algorithm requires less than 4% of the vehicle's trajectory time.

7 Discussion

This paper describes a modified FastSLAM algorithm that is uniformly superior to the FastSLAM algorithms proposed in [15]. The new FastSLAM algorithm utilizes a different proposal distribution which incorporates the most recent measurement in the pose prediction process. In doing so, it makes more efficient use of the particles, particularly in situations in which the motion noise is high in relation to the measurement noise.

A main contribution of this paper is a convergence proof for FastSLAM with a single particle. This proof is an improvement over previous formal results, which applied to algorithms much less efficient than the current one. In fact, this result is a first convergence result for a constant time SLAM algorithm.

The theoretical finding is complemented by experimental results using a standard benchmark data set. The new algorithm is found to outperform the previous FastSLAM algorithm and the EKF approach to SLAM by a large margin. In fact, a single particle suffices to generate an accurate map of a challenging benchmark data set. Despite this surprising result, the use of multiple particles is clearly warranted in situations with ambiguous data association. We believe that our results illustrate that SLAM can be solved robustly by algorithms that are significantly more efficient than EKF-based algorithms.

8 Acknowledgments

This work was supported by DARPA's MARS and MARS2020 program. We thank the Hertz Foundation for their support of Michael Montemerlo's graduate research.

References

[1] T. Bailey. *Mobile Robot Localisation and Mapping in Extensive Outdoor Environments*. PhD thesis, Univ. of Sydney, 2002.

[2] Y. Bar-Shalom and T. E. Fortmann. *Tracking and Data Association*. Academic Press, 1988.

[3] SLAM summer school 2002 `http://www.cas.kth.se/SLAM/`

[4] N. de Freitas, M. Niranjan, A. Gee, and A. Doucet. Sequential monte carlo methods to train neural network models. *Neural Computation*, 12(4), 2000.

[5] G. Dissanayake, P. Newman, S. Clark, H.F. Durrant-Whyte, and M. Csorba. A solution to the simultaneous localisation and map building (SLAM) problem. *IEEE Trans. Robotics and Automation*, 2001.

[6] A. Doucet, J.F.G. de Freitas, and N.J. Gordon, editors. *Sequential Monte Carlo Methods In Practice*. Springer, 2001.

[7] J. Guivant and E. Nebot. Optimization of the simultaneous localization and map building algorithm for real time implementation. *Trans. of Robotics and Automation*, 2001.

[8] J.-S. Gutmann and K. Konolige. Incremental mapping of large cyclic environments. *Proc. CIRA*, 2000.

[9] S. J. Julier and J. K. Uhlmann. A new extension of the Kalman filter to nonlinear systems. *Proc. AeroSense*, 1997.

[10] K. Kanazawa, D. Koller, and S.J. Russell. Stochastic simulation algorithms for dynamic probabilistic networks. *UAI-95*.

[11] *Notes ICRA Workshop Concurrent Mapping and Localization for Autonomous Mobile Robots*, 2002

[12] J.J. Leonard and H.J.S. Feder. A computationally efficient method for large-scale concurrent mapping and localization. *Proc. IRSS*, 1999.

[13] P. Maybeck. *Stochastic Models, Estimation, and Control, Volume 1*. Academic Press, 1979.

[14] M. Montemerlo and S. Thrun. Simultaneous localization and mapping with unknown data association using FastSLAM. *Proc. ICRA*, 2003, to appear.

[15] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM: A factored solution to the simultaneous localization and mapping problem. *Proc. AAAI*, 2002.

[16] H. P. Moravec. Sensor fusion in certainty grids for mobile robots. *AI Magazine*, 9(2), 1988.

[17] K. Murphy. Bayesian map learning in dynamic environments. *Proc. NIPS*, 1999.

[18] P. Newman. *On the Structure and Solution of the Simultaneous Localisation and Map Building Problem*. PhD thesis, Univ. of Sydney, 2000.
[19] M.A. Paskin. Thin junction tree filters for simultaneous localization and mapping. TR UCB/CSD-02-1198, UC Berkeley, 2002.
[20] D.B. Rubin. Using the SIR algorithm to simulate posterior distributions. In *Bayesian Statistics 3*. Oxford Univ. Press, 1988.
[21] R. C. Smith and P. Cheeseman. On the representation and estimation of spatial uncertainty. *Int. J. Robotics Research*, 5(4), 1986.
[22] J.D. Tardós, J. Neira, P.M. Newman, and J.J. Leonard. Robust mapping and localization in indoor environments using sonar data. *Int. J. Robotics Research*, 21(4), 2002.
[23] S. Thrun, D. Koller, Z. Ghahramani, H. Durrant-Whyte, and A.Y. Ng. Simultaneous mapping and localization with sparse extended information filters. *Proc. WAFR*, 2002.
[24] R. van der Merwe, N. de Freitas, A. Doucet, and E. Wan. The unscented particle filter. *Proc. NIPS*, 2001.
[25] S.B. Williams, G. Dissanayake, and H. Durrant-Whyte. An efficient approach to the simultaneous localisation and mapping problem. *Proc. ICRA*, 2002.

Appendix

The linear form (23) and (24) implies $\hat{s}_t^{[m]} = s_{t-1}^{[m]} + u_t$, $\hat{z}_t^{[m]} = \mu_{n_t,t-1}^{[m]} - s_{t-1}^{[m]} - u_t$, $G_\theta = I$, $G_s = -I$, and $Q_t^{[m]} = R_t + \Sigma_{n_t,t-1}^{[m]}$. From that we obtain for the mean and covariance (13) and (14) of the proposal distribution:

$$\Sigma_{s_t}^{[m]} = \left[(R_t + \Sigma_{n_t,t-1}^{[m]})^{-1} + P_t^{-1} \right]^{-1} \quad (25)$$

$$\mu_{s_t}^{[m]} = -\Sigma_{s_t}^{[m]} (R_t + \Sigma_{n_t,t-1}^{[m]})^{-1} (z_t - \mu_{n_t,t-1}^{[m]} + s_{t-1}^{[m]} + u_t) + s_{t-1}^{[m]} + u_t \quad (26)$$

The update of the landmark mean (Eq. (16) and (17)) resolves to:

$$\mu_{n_t,t}^{[m]} = \mu_{n_t,t-1}^{[m]} + \Sigma_{n_t,t-1}^{[m]} (R_t + \Sigma_{n_t,t-1}^{[m]})^{-1} (z_t - \mu_{n_t,t-1}^{[m]} + s_{t-1}^{[m]} + u_t) \quad (27)$$

We define the error in the robot pose and landmark locations as:

$$\alpha_t^{[m]} = s_t^{[m]} - s_t \quad \text{and} \quad \beta_{n,t}^{[m]} = \mu_{n,t}^{[m]} - \theta_n \quad (28)$$

We first characterize the effect of map errors β on the pose error α:

Lemma 1. *If the error $\beta_{n_t,t}^{[m]}$ of the observed landmark z_t at time t is smaller in magnitude than the robot pose error $\alpha_t^{[m]}$, $\alpha_t^{[m]}$ shrinks in expectation as a result of this measurement. Conversely, if $\beta_{n_t,t}^{[m]}$ is larger than the pose error $\alpha_t^{[m]}$, the latter may increase, but in expectation will not exceed $\beta_{n_t,t}^{[m]}$.*

Proof. The expected error of the robot pose at time t is given by

$$E[\alpha_t^{[m]}] = E[s_t^{[m]} - s_t] = E[s_t^{[m]}] - E[s_t] \quad (29)$$

The first term is obtained via the sampling distribution (26), and the second term is obtained from the linear motion model (24), giving:

$$E[\alpha_t^{[m]}] = -\Sigma_{s_t}^{[m]} (R_t + \Sigma_{n_t,t-1}^{[m]})^{-1} (E[z_t] - \mu_{n_t,t-1}^{[m]} + s_{t-1}^{[m]} + u_t) + \alpha_{t-1}^{[m]} \quad (30)$$

For linear SLAM, the expectation $E[z_t] = \theta_{n_t} - E[s_t] = \theta_{n_t} - u_t - s_{t-1}$. With that, the expression in the brackets becomes

$$\begin{aligned} & E[z_t] - \mu_{n_t,t-1}^{[m]} + s_{t-1}^{[m]} + u_t \\ &= \theta_{n_t} - u_t - s_{t-1} - \mu_{n_t,t-1}^{[m]} + s_{t-1}^{[m]} + u_t \\ &= \alpha_{t-1}^{[m]} - \beta_{n_t,t-1}^{[m]} \end{aligned} \quad (31)$$

Substituting this back into (30) and subsequently substituting $\Sigma_{s_t}^{[m]}$ according to (25) gives us:

$$\begin{aligned} & E[\alpha_t^{[m]}] \\ &= \alpha_{t-1}^{[m]} + \Sigma_{s_t}^{[m]} (R_t + \Sigma_{n_t,t-1}^{[m]})^{-1} (\beta_{n_t,t-1}^{[m]} - \alpha_{t-1}^{[m]}) \\ &= \alpha_{t-1}^{[m]} + \left[I + (R_t + \Sigma_{n_t,t-1}^{[m]}) P_t^{-1} \right]^{-1} (\beta_{n_t,t-1}^{[m]} - \alpha_{t-1}^{[m]}) \end{aligned} \quad (32)$$

The lemma follows from the fact that R_t, $\Sigma_{n_t,t-1}^{[m]}$, and P_t^{-1} are positive semidefinite, hence the inverse of $I + (R_t + \Sigma_{n_t,t-1}^{[m]}) P_t^{-1}$ is a contraction matrix. $E[\alpha_t^{[m]}]$ is larger in magnitude if and only if $\alpha_{t-1}^{[m]}$ depends on the sign of $\beta_{n_t,t-1}^{[m]} > \alpha_{t-1}^{[m]}$; however, $E[\alpha_t^{[m]}]$ cannot exceed $\beta_{n_t,t-1}^{[m]}$ in this case. *qed.*

Of particular interest is the result of observing the *anchoring landmark*, by which we mean the landmark whose location is known. Without loss of generality, we assume that this landmark is θ_1.

Lemma 2. *If the robot observes the anchoring landmark θ_1, its pose error will shrink in expectation.*

Proof. The anchoring landmark has zero error: $\beta_{1,t}^{[m]} = 0$, and its covariance is also zero: $\Sigma_{1,t}^{[m]} = 0$. Plugging this into (32), we get:

$$\begin{aligned} E[\alpha_t^{[m]}] &= \alpha_{t-1}^{[m]} + \left[I + (R_t + 0) P_t^{-1} \right]^{-1} (0 - \alpha_{t-1}^{[m]}) \\ &= \alpha_{t-1}^{[m]} - \left[I + R_t P_t^{-1} \right]^{-1} \alpha_{t-1}^{[m]} \end{aligned} \quad (33)$$

qed.

Finally, a lemma similar to Lemma 1 can be stated on the effect of pose errors α on map errors β. Its proof is analogous that of Lemma 1, with reverse roles of α and β.

Lemma 3. *If the pose error $\alpha_{t-1}^{[m]}$ is smaller than the error $\beta_{n_t,t}^{[m]}$ of the observed landmark z_t in magnitude, observing z_t shrinks the landmark error $\beta_{n_t,t}^{[m]}$ in expectation. Conversely, if $\alpha_{t-1}^{[m]}$ is larger than the landmark error $\beta_{n_t,t}^{[m]}$, the latter may increase, but in expectation will not exceed $\alpha_{t-1}^{[m]}$.*

Proof of Theorem. Let $\hat{\beta}_t^{[m]}$ denote landmark error that is largest in magnitude among all landmark errors at time t.

$$\hat{\beta}_t^{[m]} = \operatorname*{argmax}_{\beta_{n,t}^{[m]}} |\beta_{n,t}^{[m]}| \quad (34)$$

Lemma 3 suggests that this error may increase in expectation, but only if the absolute robot pose error $\alpha_{t-1}^{[m]}$ exceeds this error in magnitude. However, in expectation this will only be the case for a limited number of iterations. In particular, Lemma 1 guarantees that $\alpha_{t-1}^{[m]}$ may only shrink in expectation. Furthermore, Lemma 2 states that every time the anchoring landmark is observed, this error will shrink by a finite amount, regardless of the magnitude of $\hat{\beta}_t^{[m]}$. Hence, $\alpha_{t-1}^{[m]}$ will ultimately become smaller in magnitude (and in expectation) than the largest landmark error. Once this has happened, Lemma 3 states that the latter will shrink in expectation every time the landmark is observed whose error is largest. It is now easy to see that both $\hat{\beta}_t^{[m]}$ and $\alpha_{t-1}^{[m]}$ converge to zero: Observing the anchoring landmark induces a finite reduction as stated in (33). To increase $\alpha_{t-1}^{[m]}$ to its old value in expectation, the total landmark error must shrink in expectation (Lemma 3). This leads to an eternal shrinkage of the total landmark error down to zero. Since this error is an upper bound for the expected pose error (see Lemma 1), we also have convergence in expectation for the robot pose error. *qed.*