Unscented FastSLAM: A Robust and Efficient Solution to the SLAM Problem

Chanki Kim, *Student Member, IEEE*, Rathinasamy Sakthivel, and Wan Kyun Chung, *Member, IEEE*

***Abstract*—The Rao–Blackwellized particle filter (RBPF) and FastSLAM have two important limitations, which are the derivation of the Jacobian matrices and the linear approximations of nonlinear functions. These can make the filter inconsistent. Another challenge is to reduce the number of particles while maintaining the estimation accuracy. This paper provides a robust new algorithm based on the scaled unscented transformation called unscented FastSLAM (UFastSLAM). It overcomes the important drawbacks of the previous frameworks by directly using nonlinear relations. This approach improves the filter consistency and state estimation accuracy, and requires smaller number of particles than the FastSLAM approach. Simulation results in large-scale environments and experimental results with a benchmark dataset are presented, demonstrating the superiority of the UFastSLAM algorithm.**

***Index Terms*—FastSLAM, improved proposal distribution, Jacobian, linearization, Rao–Blackwellized particle filter (RBPF), scaled unscented transformation (SUT), simultaneous localization and mapping (SLAM), unscented Kalman filter (UKF), unscented particle filter (UPF).**

I. Introduction

SIMULTANEOUS localization and mapping (SLAM) is the problem of determining the position and the heading of an autonomous vehicle moving through an unknown environment, and simultaneously, of estimating the features of the environment. In the past decade, a lot of researchers have focused on the real-time SLAM problem (see [1] and [2], and references therein) and also the consistency of the SLAM algorithm [3]–[9]. Also, there has been ongoing research on issues such as resolution of nonlinearity, non-Gaussian distributions, data association, and landmark characterization, all of which are vital in achieving a practical and robust SLAM implementation.

Recently, estimation algorithms have been roughly classified according to their underlying basic principle. The most popular approaches to the SLAM problem are the extended Kalman filter (EKF-SLAM) and the Rao–Blackwellized particle filter (RBPF-SLAM) [1]. The effectiveness of the EKF approach comes from the fact that it estimates a fully correlated posterior over feature maps and vehicle poses. EKF-SLAM permits linear approximations of the motion and the measurement models, and it assumes Gaussian representations for the probability density functions. As proved in the previous research [3], [4], [7], [10], the solution of the EKF-SLAM is inconsistent due to errors introduced during linearization, which induces inaccurate maps with filter divergence. Therefore, the consistency issue of the EKF-SLAM has attracted the attention of the research community due to its importance, and many recent research efforts have concentrated on improving the classical algorithm [3], [9], [11], [12].

As a solution for this linearization issue, Martinez-Cantin and Castellanos [3] introduced the unscented Kalman filter (UKF) [13]–[16] in SLAM problems for large-scale outdoor environments. This approach was used to avoid the analytical linearization based on Taylor series expansion of the nonlinear models, and improved the consistency over the EKF-based approach. Also, Merwe *et al.* [17] applied the UKF in a particle filter algorithm for a novel proposal distribution in the sampling step. The unscented filter employs a deterministic sampling approach to capture the mean and the covariance estimates with a minimal set of sample points called *sigma points*. As proved in the previous research, the number of *sigma points* precisely and accurately approximates the posterior covariance up to the third order, whereas the EKF relies on a first-order approximation [15], [17].

Murphy [18] introduced the RBPF as an effective approach for learning grid maps by decoupling the state of the vehicle and the map. The RBPF-SLAM algorithms, with their flexibility to handle nonlinearity and non-Gaussianity, have become exceptionally popular due to their relative simplicity, computational efficiency, and experimental successes. One of the major challenges for the RBPFs is to reduce the number of particles while maintaining the estimation accuracy. Furthermore, since a resampling process of the particle filter can eliminate the correct particles [17] and can prevent maintenance of the particle diversity, determining a novel resampling technique is also important. Additionally, the consistency of the particle-filter-based SLAM algorithm became an issue of recent robotics papers [8], [19], [20].

Based on the RBPF framework, Montemerlo *et al.* [21], [22] developed a SLAM framework called FastSLAM for feature-based maps. The FastSLAM approach is summarized in Table I. The FastSLAM1.0 algorithm uses the general particle filter to estimate the vehicle pose, and each particle has an associated set of independent EKF to estimate the position of each feature in the map. FastSLAM2.0 linearizes the nonlinear model in the

Manuscript received May 21, 2007; revised October 24, 2007. This paper was recommended for publication by Associate Editor L. Parker and Editor S. Roumeliotis upon evaluation of the reviewers' comments. This work was supported in part by the Korea Science and Engineering Foundation (KOSEF) under Korea Government Grant MOST R0A-2003-000-10308-0, in part by the Korea Health 21 R&D Project, Ministry of Health and Welfare, Republic of Korea under Grant A020603, and in part by the Agency for Defence Development and by Unmanned Technology Research Center (UTRC), Korea Advanced Institute of Science and Technology.

The authors are with the Department of Mechanical Engineering, Pohang University of Science and Technology (POSTECH), Pohang 790784, South Korea (e-mail: minekiki@postech.ac.kr; sakthi@postech.ac.kr; wkchung@postech.ac.kr).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TRO.2008.924946

TABLE I
RAO–BLACKWELLIZED PARTICLE FILTER SLAM

	vehicle state	feature states
FastSLAM1.0	general particle filter	EKFs with **linear approx.**
FastSLAM2.0	proposal distribution from **linear approx.**	EKFs with **linear approx.**
UFastSLAM	proposal distribution from unscented filter **without** linear approx.	unscented filters **without** linear approx.

same manner as the EKF-SLAM to improve the accuracy of a proposal distribution, and it also employs the low-dimensional EKFs for estimating feature states.

However, the FastSLAM and some EKF-based RBPF frameworks have two important potential drawbacks, which are the derivation of the Jacobian matrices and the linear approximations of the nonlinear functions. Calculating the Jacobian is unwelcome effort, and inaccurate approximation to the posterior covariance degenerates the estimate accuracy and the filter consistency.

This paper proposes a new probabilistic framework called unscented FastSLAM (UFastSLAM) to overcome the drawbacks caused by linearizations in the FastSLAM framework. The linearization process with Jacobian calculations is removed by applying the scaled unscented transformation (SUT) [16] to the factorized SLAM framework. Our main contributions are as follows.

1) UFastSLAM, unlike FastSLAM, computes the proposal distribution by measurement updates of the unscented filter in the sampling step of the particle filter. In particular, when multiple features are observed at the same time, the *sigma points* of the unscented filter are updated in every feature update step for accurate vehicle pose estimation. This avoids the linearized transformations of the vehicle uncertainty and derivation of Jacobian matrices.
2) UFastSLAM also updates each feature state by the unscented filter without accumulating the linear approximation error and without calculating the Jacobian matrix of an observation model. In addition, the use of SUT removes the impact of an inaccurate linearized transformation from polar to Cartesian coordinates in the feature initialization.

UFastSLAM is summarized in Table I.

We observe experimentally that the proposed UFastSLAM algorithm, even with fewer particles, yields significantly more accurate estimate results when compared with FastSLAM2.0. This improvement is salient with large measurement uncertainty. Furthermore, without using linearization for both the feature and the vehicle estimate, the filter consistency of the UFastSLAM is greatly improved over the FastSLAM2.0 algorithm.

The research presented in this paper is an extension of our preliminary results presented in [23], as it further evaluates a filter consistency and analyzes a complexity. Furthermore, we added the importance weight equation, a formal description of the techniques used, and provide more detailed experiments including outdoor results as well as indoors. Especially, these real-world experiments employ both laser range finder and sonar to validate robustness to sensor noises.

We note that a similar approach has also been independently developed in [24]. A major difference with our preliminary version [23] was whether an augmented technique for feature state was adopted to update a map, and its effect on the filter is also analyzed in this paper.

The structure of the paper is as follows. Section II briefly reviews the original FastSLAM framework, and the necessary notations are made. The main contribution of this paper is shown in Section III, which presents the UFastSLAM algorithm to solve the SLAM problem. This section shows how the proposal distribution is calculated using the unscented filtering technique in the RBPF framework, explains the measurement update and initialization of the feature. Section IV gives the pseudocode of the UFastSLAM algorithm. The effectiveness of the proposed algorithm is demonstrated using simulation results and experimental results in Sections V and VI, respectively. Finally, Section VII contains the concluding remarks with a discussion of current shortcomings.

II. BACKGROUND: FASTSLAM FRAMEWORK

Before introducing FastSLAM, the SLAM problem should be presented from a probabilistic point of view. In probabilistic terms, the SLAM problem is a Markov process [25]. In particular, the vehicle motion is usually considered as a Markov process. The vehicle's pose at time t will be denoted by x_t. The vehicle's environment possesses N static landmarks. Each landmark is denoted by θ_k for $k = 1, \ldots, N$. The set of all landmarks will be denoted by θ. SLAM algorithms calculate the posterior over the entire path along with the map

$$p(x^t, \theta | z^t, u^t, n^t) \tag{1}$$

where the path of the vehicle is denoted by $x^t = x_1, \ldots, x_t$, $z^t = z_1, \ldots, z_t$ is a sequence of measurements and $u^t = u_1, \ldots, u_t$ is a sequence of control inputs. The variables $n^t = n_1, \ldots, n_t$ are data association variables, in which each n_t specifies the identity of the landmark observed at time t. The SLAM problem is simultaneously inferring the location of all landmarks θ in the environment and the path x^t followed by the vehicle, based on a set of measurements and inputs.

To compute the posterior (1), the evolution of poses according to a probabilistic law is commonly referred to as a nonlinear motion model

$$p(x_t | x_{t-1}, u_t).$$

The current pose x_t is a probabilistic function of the vehicle control u_t and the previous pose x_{t-1}. As the vehicle moves around, it takes measurements of its environment. Moreover, sensor measurements are governed by a probabilistic law referred to as a nonlinear measurement model

$$p(z_t | x_t, \theta, n_t).$$

The FastSLAM algorithm, introduced by Montemerlo *et al.* [22], is an efficient algorithm for the SLAM problem that is based on a straightforward factorization. This algorithm partitions the SLAM posterior into a localization problem and independent landmark position estimation problem conditioned on

the vehicle pose estimate. The conditional independence property of the SLAM problem enables us to estimate the posterior (1) in the following factored form:

$$p(x^t, \theta | z^t, u^t, n^t) = p(x^t | z^t, u^t, n^t) \prod_{k=1}^{N} p(\theta_k | x^t, z^t, u^t, n^t).$$

This factorization [18] is the fundamental idea behind FastSLAM.

FastSLAM uses a particle filter to approximate the ideal recursive Bayesian filter for estimating the vehicle pose. The remaining posterior of feature locations is analytically calculated by using the EKF filters. So the FastSLAM algorithm is a Rao–Blackwellized particle filter. Each particle in the FastSLAM is of the form

$$X_t^{[m]} = \left\langle x^{t,[m]}, \mu_{1,t}^{[m]}, \Sigma_{1,t}^{[m]}, \ldots, \mu_{N,t}^{[m]}, \Sigma_{N,t}^{[m]} \right\rangle$$

where $[m]$ indicates the index of the particle, $x^{t,[m]}$ is its path estimate, and $\mu_{k,t}^{[m]}$ and $\Sigma_{k,t}^{[m]}$ are the mean and covariance of the Gaussian, representing the kth landmark location that is attached to the mth particle. In FastSLAM1.0, new poses are sampled using the most recent motion command u_t

$$x_t^{[m]} \sim p\left(x_t | x_{t-1}^{[m]}, u_t\right).$$

It is important to note that this proposal distribution uses only the motion control u_t, but ignores the current measurement z_t. So the FastSLAM1.0 approach is particularly troublesome if the observation is too accurate relative to the vehicle's motion noise. To overcome this problem, an improved version called FastSLAM2.0 is proposed by Montemerlo *et al.* [22]. In FastSLAM2.0, the vehicle poses are sampled under consideration of both the control u_t and the measurement z_t, which is denoted by the following sampling distribution:

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

and as a result, the FastSLAM2.0 is superior to the FastSLAM1.0 in all aspects [8].

The weight of each sample used in the resampling step is called the importance weight, which is denoted by $w_t^{[m]}$. In FastSLAM2.0, the importance weight for resampling is given by

$$w_t^{[m]} = \frac{\text{target distribution}}{\text{proposal distribution}} = \frac{p(x^{t,[m]} | z^t, u^t, n^t)}{p(x^{t-1,[m]} | z^{t-1}, u^{t-1}, n^{t-1}) p(x_t^{[m]} | x^{t-1,[m]}, z^t, u^t, n^t)}.$$

For a complete derivation of the importance weight in FastSLAM2.0, see [21] and [22].

In the FastSLAM algorithm, each particle does not represent a single momentary vehicle pose. Rather, it represents an entire vehicle path history and the path history is recorded in the map estimates. The map is represented as a set of independent Gaussians, with linear time complexity, rather than a joint map covariance with quadratic time complexity. Moreover, if the landmarks are represented by a binary tree, linear complexity in the number of landmarks becomes the log-time. These are key properties of the FastSLAM and are the reason for its computational speed. In addition to the obvious reduction in computational complexity, the technique intrinsically provides a way of maintaining multihypothesis data association.

In spite of these advantages, FastSLAM suffers from the drawbacks of linearization. The linearized approach can produce a critical problem in the particle filter such that the inaccurate estimation caused by the linear approximation brings the particles to converge to a wrong pose with rapid loss of the filter consistency after resampling. Using EKFs for the feature estimation also contains the linearization errors, and the error is salient with large bearing noise of a range finder sensor. This approach can prevent a successful data association and can make an inaccurate map.

To solve these problems, we propose the UFastSLAM in the following section.

III. Unscented FastSLAM

Unlike FastSLAM, UFastSLAM is based on the SUT. UFastSLAM consists of three parts: the vehicle state estimation, the feature state estimation, and the importance weights calculation.

A. Vehicle State Estimation: Sampling Strategy in UFastSLAM

The particle filter in the RBPF framework relies on importance sampling, so it requires the design of proposal distributions that can approximate the true posterior reasonably well. The most common strategy is to sample from the motion model. This strategy can fail, however, if the new measurements appear in the tail of the proposal distribution, or if the likelihood is too sharp in comparison to the proposal distribution.

Several researchers have introduced the most current observations into the proposal distribution and have used some heuristic techniques to improve the accuracy of the proposal distribution [22], [26]–[28]. However, the Jacobian and inaccurate linear approximations still exist in the covariance-related terms.

Instead of linearinzing the nonlinear models through the first-order Taylor series expansion at the mean of the vehicle state, UFastSLAM computes a more accurate mean and more precise uncertainty of the vehicle by applying the unscented particle filter (UPF) technique, which takes into account a linear regression of weighted points [29]. The original UPF [17], however, assumed that an observation is obtained in each time step. However, in the SLAM problem, a lack of an observation or multiple observations can happen frequently, so the original UPF was modified for the SLAM purpose.

Since an observation is not always detected, constructing the proposal distribution and sampling from this prior have two steps. One is the prediction step and another is the measurement update step. At first, the state vector is augmented with a control input and the observation

$$x_{t-1}^{a[m]} = \begin{bmatrix} x_{t-1}^{[m]} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} = \begin{bmatrix} x_{x,t-1}^{[m]} \\ x_{y,t-1}^{[m]} \\ x_{\theta,t-1}^{[m]} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \quad P_{t-1}^{a[m]} = \begin{bmatrix} P_{t-1}^{[m]} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & Q_t & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & R_t \end{bmatrix}. \tag{2}$$

Here, $x_{t-1}^{a[m]}$ is the augmented vector for the state and $x_{t-1}^{[m]}$ is the previous mean of the vehicle. Q_t and R_t are the control noise covariance and the measurement noise covariance, respectively. The augmented covariance matrix $P_{t-1}^{a[m]}$ has 7×7 dimensions in 3 DOF vehicle state estimation.

UFastSLAM deterministically extracts *sigma points* from the Gaussian and passes these points through the nonlinear function. A symmetric set of $2L+1$ *sigma points* $\chi_{t-1}^{a[i][m]}$ for the augmented state vector with $L=7$ can be calculated and are given by

$$\chi_{t-1}^{a[0][m]} = x_{t-1}^{a[m]}$$

$$\chi_{t-1}^{a[i][m]} = x_{t-1}^{a[m]} + \left(\sqrt{(L+\lambda)P_{t-1}^{a[m]}}\right)_i \quad (i=1,\ldots,L)$$

$$\chi_{t-1}^{a[i][m]} = x_{t-1}^{a[m]} - \left(\sqrt{(L+\lambda)P_{t-1}^{a[m]}}\right)_{i-L} \quad (i=L+1,\ldots,2L)$$

where subscript i means the ith column of a matrix. The λ is computed by $\lambda = \alpha^2(L+\kappa) - L$ and α $(0 < \alpha < 1)$ should be a small number to avoid sampling nonlocal effects when the nonlinearities are strong ($\alpha = 0.002$ is appropriate). κ is a scaling parameter that determines how far the sigma points are separated from the mean. The specific value of κ ($\kappa \geq 0$ to guarantee positive semidefiniteness of the covariance matrix) is not critical though, so a good default choice is $\kappa = 0$. Each *sigma point* $\chi_{t-1}^{a[i][m]}$ contains the state, control, and measurement components given by

$$\chi_{t-1}^{a[i][m]} = \begin{bmatrix} \chi_{t-1}^{[i][m]} \\ \chi_t^{u[i][m]} \\ \chi_t^{z[i][m]} \end{bmatrix}. \tag{3}$$

The motion model $\mathbf{f}$ is characterized by a nonlinear function, and the set of *sigma points* $\chi_{t-1}^{a[i][m]}$ is transformed by the motion model using the current control $u_t^{[m]}$ with the added control noise component $\chi_t^{u[i][m]}$ of each *sigma point*

$$\bar{\chi}_t^{[i][m]} = \mathbf{f}\left(u_t^{[m]} + \chi_t^{u[i][m]}, \chi_{t-1}^{[i][m]}\right) = \begin{bmatrix} \bar{\chi}_{x,t}^{[i][m]} \\ \bar{\chi}_{y,t}^{[i][m]} \\ \bar{\chi}_{\theta,t}^{[i][m]} \end{bmatrix}. \tag{4}$$

Here, $\bar{\chi}_t^{[i][m]}$ is the transformed *sigma point* of the vehicle state. [See Section VII-A for detailed expression of (4).] The first two moments of the predicted vehicle state are computed by a linear weighted regression of the transformed *sigma points* $\bar{\chi}_t^{[i][m]}$:

$$x_{t|t-1}^{[m]} = \sum_{i=0}^{2L} w_g^{[i]} \bar{\chi}_t^{[i][m]} \tag{5}$$

$$P_{t|t-1}^{[m]} = \sum_{i=0}^{2L} w_c^{[i]} \left(\bar{\chi}_t^{[i][m]} - x_{t|t-1}^{[m]}\right)\left(\bar{\chi}_t^{[i][m]} - x_{t|t-1}^{[m]}\right)^T \tag{6}$$

where a weight $w_g^{[i]}$ is used when computing the mean, and the weight $w_c^{[i]}$ is used when recovering the covariance of the Gaussian. These weights are calculated by

$$w_g^{[0]} = \frac{\lambda}{(L+\lambda)}, \quad w_c^{[0]} = \frac{\lambda}{(L+\lambda)} + (1-\alpha^2+\beta)$$

$$w_g^{[i]} = w_c^{[i]} = \frac{1}{2(L+\lambda)} \quad (i=1,\ldots,2L). \tag{7}$$

Here, the parameter β is used to incorporate the knowledge of the higher order moments of the posterior distribution. For a Gaussian prior, the optimal choice is $\beta = 2$ [16], [25].

As some features are observed, data association provides their identities, and (8)–(14) are employed to update the estimated mean $x_t^{[m]}$ and the covariance $P_t^{[m]}$ of the vehicle:

$$\bar{N}_t^{[i][m]} = \mathbf{h}\left(\bar{\chi}_t^{[i][m]}, \mu_{\hat{k},t-1}^{[m]}\right) + \chi_t^{z[i][m]} \tag{8}$$

$$\hat{n}_t^{[m]} = \sum_{i=0}^{2L} w_g^{[i]} \bar{N}_t^{[i][m]} \tag{9}$$

$$S_t^{[m]} = \sum_{i=0}^{2L} w_c^{[i]} \left(\bar{N}_t^{[i][m]} - \hat{n}_t^{[m]}\right)\left(\bar{N}_t^{[i][m]} - \hat{n}_t^{[m]}\right)^T \tag{10}$$

$$\Sigma_t^{x,n[m]} = \sum_{i=0}^{2L} w_c^{[i]} \left(\bar{\chi}_t^{[i][m]} - x_{t|t-1}^{[m]}\right)\left(\bar{N}_t^{[i][m]} - \hat{n}_t^{[m]}\right)^T \tag{11}$$

$$K_t^{[m]} = \Sigma_t^{x,n[m]} \left(S_t^{[m]}\right)^{-1}. \tag{12}$$

The measurement *sigma points* $\bar{N}_t^{[i][m]}$ are calculated in (8) using the observation model $\mathbf{h}$, characterized by a nonlinear function, with the added measurement noise component $\chi_t^{z[i][m]}$. Here, $\mu_{\hat{k},t-1}^{[m]}$ is the previously registered $\hat{k}$th feature mean that is reobserved and identified in the current time step. The difference between the reobserved feature position and the *sigma point* of the vehicle pose $\bar{\chi}_t^{[i][m]}$ is used to calculate the measurement *sigma points* $\bar{N}_t^{[i][m]}$. $\hat{n}_t^{[m]}$ is the predicted measurement and $S_t^{[m]}$ is the innovation covariance. The cross-covariance $\Sigma_t^{x,n[m]}$ corresponds to the Jacobian term of the FastSLAM algorithm. $K_t^{[m]}$ is the Kalman gain in the measurement update.

Note that the measurement noise covariance R_t is not used as an additive term for calculating the innovation covariance $S_t^{[m]}$. This is because the measurement noise is already included in the augmented covariance (2) and was already considered to calculate the predicted measurement $\hat{n}_t^{[m]}$. Alternatively, the augmented state vector and the augmented covariance (2) could be defined using the state and control inputs, and rather than containing the measurement term in the augmentation, the measurement noise covariance could be handled as an additive term in (10). However, it is important to note that this can lead to a negative definite of the covariance of the vehicle $P_t^{[m]}$ for a very small measurement noise with multiple observations.

The estimated mean and its covariance of the vehicle state at time t are calculated by

$$x_t^{[m]} = x_{t|t-1}^{[m]} + K_t^{[m]}\left(z_t - \hat{n}_t^{[m]}\right) \tag{13}$$

$$P_t^{[m]} = P_{t|t-1}^{[m]} - K_t^{[m]} S_t^{[m]} \left(K_t^{[m]}\right)^T. \tag{14}$$

From the Gaussian distribution generated by the estimated mean and covariance of the vehicle, the state of each particle is sampled:

$$x_t^{[m]} \sim \mathcal{N}\left(x_t^{[m]}, P_t^{[m]}\right). \tag{15}$$

When there is no observation, the vehicle state is predicted without the measurement update using (5) and (6), but if many features are observed at the same time, (8)–(14) are repeated for each observed feature, and the mean and the covariance of the vehicle are updated based on the previously updated one. In the multiple-observation case, before considering the next observed feature, the *sigma point* $\chi_t^{a[i][m]}$ is refreshed by using the updated mean (13) and covariance (14)

$$\chi_t^{a[i][m]} = \left[x_t^{a[m]},\ x_t^{a[m]} \pm \sqrt{(L+\lambda)P_t^{a[m]}}\right] \tag{16}$$

$$\bar{\chi}_t^{[i][m]} = \chi_t^{[i][m]}. \tag{17}$$

This approach produces accurate *sigma points* in each update step and is important in estimating the vehicle state accurately.

The effectiveness of this approach to the vehicle state estimation is illustrated in Section V-A, where it is called "modified UPF aided FastSLAM."

B. Feature State Estimation

1) Feature Update: The feature update defines the *sigma points* using the previously registered mean and covariance of the feature.

$$\chi^{[0][m]} = \mu_{n_t,t-1}^{[m]}$$

$$\chi^{[i][m]} = \mu_{n_t,t-1}^{[m]} + \left(\sqrt{(n+\lambda)\Sigma_{n_t,t-1}^{[m]}}\right)_i (i=1,\ldots,n) \tag{18}$$

$$\chi^{[i][m]} = \mu_{n_t,t-1}^{[m]} - \left(\sqrt{(n+\lambda)\Sigma_{n_t,t-1}^{[m]}}\right)_{i-n} (i=n+1,\ldots,2n)$$

where $\mu_{n_t,t-1}^{[m]}$ is the mean of the nth feature that is registered in feature initialization step. $\Sigma_{n_t,t-1}^{[m]}$ is the covariance matrix of the nth feature, and it has 2×2 dimensions in the RBPF framework. n is the dimension of the feature state. If landmarks are on a planar environment, it has only 2 DOF. In this case, two-dimensional Gaussian ($n=2$) is sufficient, and five *sigma points* are required. $\lambda = \alpha^2(n+\kappa) - n$, and $\alpha = 0.01$ and $\kappa = 0$ are appropriate for estimating the feature state.

The predicted measurement $\hat{z}_t^{[m]}$ and the Kalman gain $\bar{K}_t^{[m]}$ are calculated as follows:

$$\bar{Z}_t^{[i][m]} = \mathbf{h}\left(\chi^{[i][m]}, x_t^{[m]}\right) \quad (i=0,\ldots,2n)$$

$$\hat{z}_t^{[m]} = \sum_{i=0}^{2n} w_g^{[i]} \bar{Z}_t^{[i][m]}$$

$$\bar{S}_t^{[m]} = \sum_{i=0}^{2n} w_c^{[i]} \left(\bar{Z}_t^{[i][m]} - \hat{z}_t^{[m]}\right)\left(\bar{Z}_t^{[i][m]} - \hat{z}_t^{[m]}\right)^T + R_t.$$

Here, $\mathbf{h}(\cdot)$ is the observation model and the current vehicle state of the mth particle $x_t^{[m]}$ is used. The transformed *sigma points* $\bar{Z}_t^{[i][m]}$ are on the polar coordinate made by the nonlinear transformation and are used to compute the predicted measurement $\hat{z}_t^{[m]}$ and the innovation covariance $\bar{S}_t^{[m]}$. The weights $w_g^{[i]}$ and $w_c^{[i]}$ are calculated as in the previous step using (7).

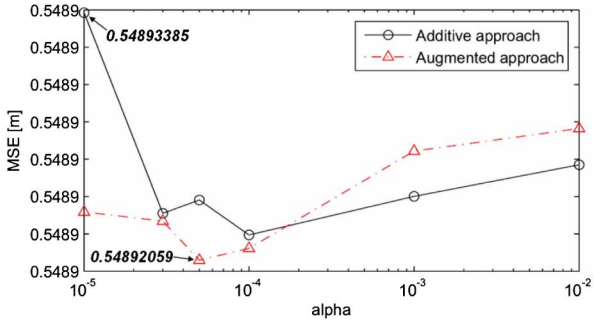


Fig. 1. Performance of the additive and the augmented approaches for the measurement noise term in the feature update. Large measurement noises are used (especially a large bearing noise of 8°) to confirm the difference between the approaches.

The measurement noise covariance R_t is used as an additive term for calculating the innovation covariance $\bar{S}_t^{[m]}$. An alternative way to update the feature is using the augmented state with the measurement noise term, given in Section VII-B. The difference between the additive approach and the augmented approach was verified under the same carefully prepared experimental conditions with large measurement noise values. The performance of both approaches are shown in Fig. 1. As shown in Fig. 1, the performance of each approach varies according to the α-value, but the difference of the MSE is small enough to be neglected. This is because the feature state has a low-dimensional, constant size in UFastSLAM. However, the augmented approach requires nine *sigma points*, whereas the additive approach requires only five *sigma points*. Thus, considering the measurement noise as an additive term is more efficient for feature estimation.

$\bar{\Sigma}_t^{[m]}$ determines the cross-covariance between state and observation, which is used to compute the Kalman gain $\bar{K}_t^{[m]}$, and is calculated by

$$\bar{\Sigma}_t^{[m]} = \sum_{i=0}^{2n} w_c^{[i]} \left(\chi^{[i][m]} - \mu_{n_t,t-1}^{[m]}\right)\left(\bar{Z}_t^{[i][m]} - \hat{z}_t^{[m]}\right)^T$$

$$\bar{K}_t^{[m]} = \bar{\Sigma}_t^{[m]} \left(\bar{S}_t^{[m]}\right)^{-1}. \tag{19}$$

The Kalman gain calculated through UFastSLAM is more accurate than the FastSLAM one. Finally, the mean $\mu_{n_t,t}^{[m]}$ and the covariance $\Sigma_{n_t,t}^{[m]}$ of the nth feature are updated by

$$\mu_{n_t,t}^{[m]} = \mu_{n_t,t-1}^{[m]} + \bar{K}_t^{[m]}\left(z_t - \hat{z}_t^{[m]}\right) \tag{20}$$

$$\Sigma_{n_t,t}^{[m]} = \Sigma_{n_t,t-1}^{[m]} - \bar{K}_t^{[m]} \bar{S}_t^{[m]} \left(\bar{K}_t^{[m]}\right)^T \tag{21}$$

where z_t is the true measurement. The Cholesky factorization is used in this feature update to make the algorithm more stable numerically. Also note that the UFastSLAM does not employ Jacobian matrices for calculating feature covariance.

2) Feature Initialization: In the RBPF framework, the feature initialization is separated from the vehicle state estimator. This means that it is possible to use the measurement noise covariance as an initial feature covariance [21], [22]. Therefore, the UFastSLAM feature initialization uses the current measurement z_t and measurement noise covariance R_t to calculate the *sigma points* $\psi^{[i][m]}$ with a two-dimensional Gaussian ($l = 2$). Here, $\lambda = \alpha^2(l + \kappa) - l$, and α and κ can be of the same value as the feature update. The mean $\mu_{n_t,t}^{[m]}$ and the covariance $\Sigma_{n_t,t}^{[m]}$ of a new feature are calculated by

$$\begin{aligned}
\psi^{[0][m]} &= z_t \\
\psi^{[i][m]} &= z_t + (\sqrt{(l+\lambda)R_t})_i \quad (i = 1, \ldots, l) \\
\psi^{[i][m]} &= z_t - (\sqrt{(l+\lambda)R_t})_{i-l} \quad (i = l+1, \ldots, 2l) \\
\bar{M}_t^{[i][m]} &= \mathbf{h}^{-1}\left(\psi^{[i][m]}, x_t^{[m]}\right) \quad (i = 0, \ldots, 2l) \\
\mu_{n_t,t}^{[m]} &= \sum_{i=0}^{2l} w_g^{[i]} \bar{M}_t^{[i][m]} \qquad (22) \\
\Sigma_{n_t,t}^{[m]} &= \sum_{i=0}^{2l} w_c^{[i]} \left(\bar{M}_t^{[i][m]} - \mu_{n_t,t}^{[m]}\right)\left(\bar{M}_t^{[i][m]} - \mu_{n_t,t}^{[m]}\right)^T. \qquad (23)
\end{aligned}$$

Note that no explicit calculation of Jacobians are necessary to implement the UFastSLAM algorithm. The UFastSLAM requires computation of a matrix square root that can be implemented directly using the Cholesky factorization. However, the covariance matrices have low dimensions and can be expressed recursively. Thus, not only does the UFastSLAM outperform the FastSLAM in accuracy and robustness, but it also does so at no extra computational cost.

The effectiveness of this approach to the feature state estimation is illustrated in Section V-A, where it is called "UKF aided FastSLAM2.0."

C. Calculating Importance Weight and Resampling Strategy

Like FastSLAM2.0, the importance weight of UFastSLAM should be computed by considering the most recent observations, and it is given by the following equation:

$$w_t^{[m]} = \left|2\pi L_t^{[m]}\right|^{-(1/2)} \mathrm{e}^{\left\{-(1/2)\left(z_t - \hat{z}_t^{[m]}\right)^T \left(L_t^{[m]}\right)^{-1}\left(z_t - \hat{z}_t^{[m]}\right)\right\}} \qquad (24)$$

where

$$L_t^{[m]} = \left(\Sigma_t^{x,n[m]}\right)^T \left(P_t^{[m]}\right)^{-1} \Sigma_t^{x,n[m]} + \bar{S}_t^{[m]}. \qquad (25)$$

One of the major influences on the performance of the particle filter is the resampling process. The particles with a low importance weight are replaced by samples with a high weight during the resampling process. The resampling technique used the effective number of particles [30] as a criteria. The effective number of particles is calculated by

$$N_{\mathbf{eff}} = \frac{1}{\sum_{m=1}^{M} (\hat{w}^{[m]})^2} \qquad (26)$$

where M is the total number of particles and $\hat{w}^{[m]}$ is the normalized weight of the mth particle. If the variance of the importance weights increases, the $N_{\mathbf{eff}}$ decreases. When $N_{\mathbf{eff}}$ drops below a threshold of 50% of the total number of particles, resampling occurs. This can handle the degeneracy caused by the variance increase.

The performance and effect of the aforementioned three parts will be shown in Section V.

D. Complexity of UFastSLAM

This section describes the complexity of UFastSLAM. Since the sigma point and linear regression calculations have constant complexities, the number of particles M determines the complexity of the filter. In UFastSLAM, computing the proposal distribution, updating the landmarks, and calculating the importance weights have a complexity of $O(M)$. On the other hand, resampling requires a complexity of $O(MN)$ [or $O(M \log N)$, if the landmarks are represented by a binary tree]. To conclude, UFastSLAM has the same complexity as FastSLAM.

IV. UFastSLAM Overview

The pseudocode of UFastSLAM is as follows.

Unscented FastSLAM $(z_t, u_t, \mathbf{X_{t-1}}, Q_t, R_t)$:
$n =$ feature dimension; $L =$ vehicle dimension
Set SUT parameters $(\alpha, \beta, \kappa, \lambda)$
Calculate SUT weights $(w_g^{[i]}, w_c^{[i]}, i = 0 \sim 2n(\text{or } 2L))$
$\mathbf{X_t} = \mathbf{X_{aux}} = \emptyset$
for all particles
  Retrieve $\mathbf{X_{t-1}}$
  Predict mean and covariance of the vehicle (5) and (6)
  for all observations
    $\hat{k} =$ compatibility test$\left(z_t, \langle \mu_{k,t-1}^{[m]}, \Sigma_{k,t-1}^{[m]} \rangle\right)$
  end for
  for $\hat{k} =$ known feature
    Update mean and covariance of the vehicle (13) and (14)
    Refresh sigma points (16) and (17)
    $w_t^{[m]} =$ calculate importance weight (24)
  end for
  Sample from updated posterior (15)
  if $\hat{k} =$ new feature
    Calculate new feature mean and covariance (22) and (23)
  else
    Update mean and covariance of feature (20) and (21)
  end if
  for unobserved features
    $\mu_{k,t}^{[m]} = \mu_{k,t-1}^{[m]} \quad \Sigma_{k,t}^{[m]} = \Sigma_{k,t-1}^{[m]}$
  end for
  add $\left\langle x_t^{[m]}, N_t^{[m]}, \left\langle \mu_{N_t^{[m]},t}^{[m]}, \Sigma_{N_t^{[m]},t}^{[m]} \right\rangle \right\rangle$ **to** $\mathbf{X_{aux}}$
end for
Resample from $\mathbf{X_{aux}}$ with probability $\propto w_t^{[m]}$
Add new particles to $\mathbf{X_t}$
Return $\mathbf{X_t}$

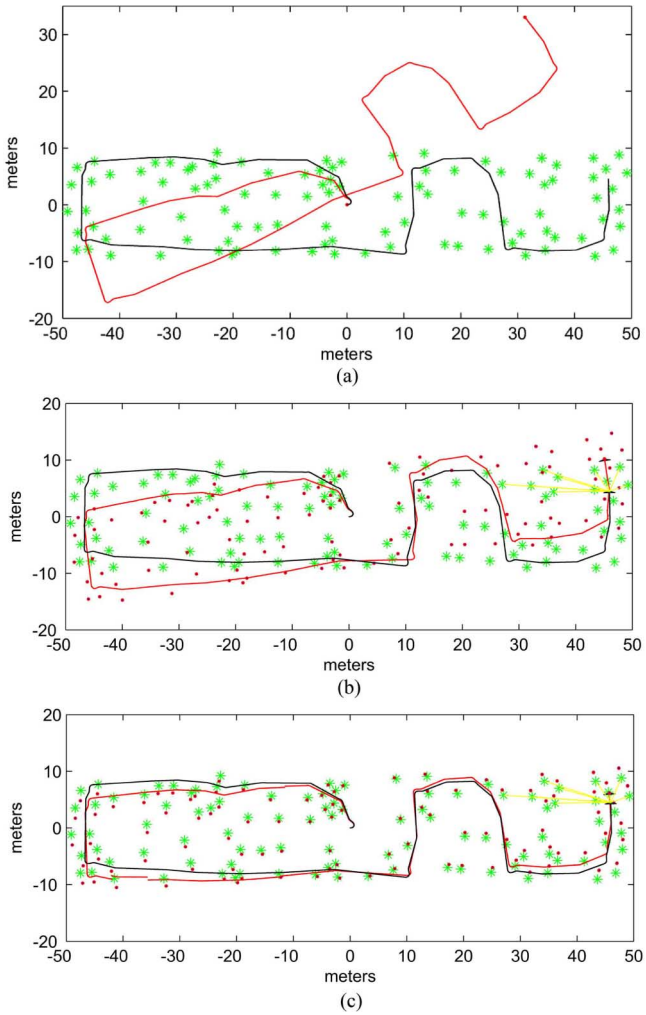


Fig. 2. Estimated and true vehicle paths with estimated and true landmarks. (a) Odometry path. (b) FastSLAM2.0. (c) UFastSLAM. The black line and green stars denote the true path and landmark positions, respectively. The red line is the mean estimate of vehicle position and the red dots are estimated landmarks. The measurement noises are relatively large ($\sigma_r = 0.2\,\mathrm{m}$, $\sigma_\phi = 8^\circ$) and only three particles were used.

V. Simulation Results

Bailey *et al.* developed the SLAM simulator and opened it to the public on a Web site [31]. This simulator made the comparison of different SLAM algorithms possible, and Bailey *et al.* [8] discussed the consistency of the FastSLAM2.0 using this simulator. UFastSLAM was developed and compared with the FastSLAM2.0 code for the same environment.

Before comparing UFastSLAM with FastSLAM2.0, results of two important versions of the improved FastSLAM algorithms are shown: the UKF-aided FastSLAM2.0 and the modified UPF-aided FastSLAM. UKF-aided FastSLAM2.0 employs the same sampling technique as FastSLAM2.0 but feature states are updated using the UKF approach and initialized by the SUT. On the other hand, modified UPF-aided FastSLAM consists of the unscented-filter-based sampling technique and the standard EKF for the feature estimation. These approaches were described in Section III-A and III-B.

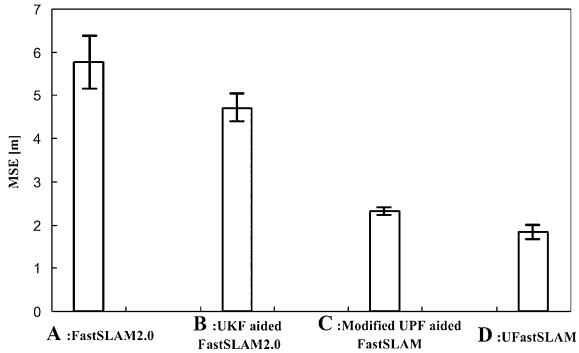


Fig. 3. Comparison of the developed algorithms with FastSLAM2.0.

TABLE II
Preliminary Experiments

SLAM algorithm (with 3 particles)	MSE [m] mean	MSE [m] var	heading errors [deg]
FastSLAM2.0	5.783	0.500	-6.6927
UKF aided FastSLAM2.0	4.705	0.352	-5.4442
Modified UPF aided FastSLAM	2.322	0.001	-2.9198
Unscented FastSLAM	1.831	0.076	-2.4861

A. Preliminary Experiments

The preliminary experiment was executed on rectangular shaped trajectories of 100 m $\times$ 20 m, as shown in Fig. 2. The vehicle has a 0.26-m wheel base and is equipped with a range-bearing sensor with a maximum range of 20 m and a 180° frontal field-of-view. Gaussian noise covariances are generated for both the measurement and the motion. The control frequency was 40 Hz and observation scans were obtained every 5 Hz. Data association is assumed to be known. The measurement noise was 0.2 m in range and 8° in bearing. We used three particles to observe the estimate accuracy of each particle. The feature and the vehicle states were predicted and updated by unscented transformation in UFastSLAM, which outperforms the FastSLAM2.0, as shown in Fig. 2.

Each bar in Fig. 3 represents the mean of the estimated error of the vehicle pose. The mean and variance of the MSE are calculated over ten independent runs for each algorithm.

In Fig. 3 and Table II, the estimation accuracy of the UKF-aided FastSLAM2.0 (**B** in Fig. 3) is better than that of FastSLAM2.0 (**A** in Fig. 3). This is because a more accurate transformation in the estimator produces better estimate results than the linear approximation of the nonlinear function. However, the reduction of the MSE is only 1 m. This means that the effect of the unscented filter in the UKF-aided FastSLAM2.0 (**B** in Fig. 3) is not so large. This is because the feature covariance has a low dimension in the RBPF framework.

The most important factor of the particle filter approach is the sampling process. The FastSLAM framework is also a particle-filter-based estimator, and as a result, the novel sampling technique affects the performance of the estimator directly. The modified UPF-aided FastSLAM (**C** in Fig. 3) had greatly increased performance when compared with the UKF-aided FastSLAM2.0 (**B** in Fig. 3).

Finally, the proposed algorithm UFastSLAM (**D** in Fig. 3) had the most accurate result.

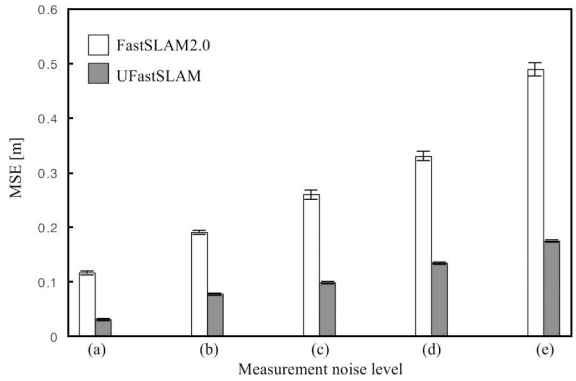


Fig. 4. Effectiveness of UFastSLAM for large measurement noises. Each bar represents the mean of the estimated vehicle pose. The mean and variance of the MSE were calculated over five independent runs for each SLAM algorithm. Ten particles were used in a small environment (20 m × 25 m). The measurement noises of each experiment are as follows. (a) $\sigma_r = 0.1, \sigma_\phi = 3^\circ$. (b) $\sigma_r = 0.2, \sigma_\phi = 6^\circ$. (c) $\sigma_r = 0.2, \sigma_\phi = 8^\circ$. (d) $\sigma_r = 0.3, \sigma_\phi = 10^\circ$. (e) $\sigma_r = 0.3, \sigma_\phi = 14^\circ$. The loop closure with known data association and the long range of an observation (20 m) induced small MSEs in all experiments.

B. Performance Comparison Between UFastSLAM and FastSLAM2.0

This section considers three types of simulation experiments to demonstrate the effectiveness of the proposed algorithm, UFastSLAM.

The first set of experiments has been designed to validate the effect of the measurement uncertainty. In this experiment, ten particles were used with various measurement noise levels in a relatively small environment (20 m × 25 m). The mean and variance of the MSE are calculated over five independent runs for each algorithm. As the measurement noise was increased, the estimation errors of the FastSLAM2.0 and the UFastSLAM algorithms increased, as shown in Fig. 4. However, the estimation error of UFastSLAM increased much more slowly than that of FastSLAM2.0, and the variance of the MSE of the UFastSLAM is smaller than that of FastSLAM2.0. Thus, UFastSLAM is more robust to high sensor noise than FastSLAM2.0.

The sensor returns polar information (range and bearing), which is converted to estimate Cartesian coordinates. In this process, the large bearing uncertainty can lead to violations of the assumption of local linearity. Therefore, the EKF's mean is biased, and even if there is no bias, the transformation of uncertainties can become inconsistent. Due to this limitation, the MSE of the FastSLAM2.0 is rapidly increased with large error variances. On the other hand, the unscented filter is able to estimate the mean and the covariance to a higher order of accuracy than the EKF. Even if there is a large bearing uncertainty, higher order information about the state distribution can be captured using a small number of *sigma points*, which produces the robustness of UFastSLAM to the sensor noise.

The second experiment is concerned with how many particles are needed in UFastSLAM to obtain the same estimation accuracy as FastSLAM2.0. The size of the environment was about 100 m × 100 m, and the vehicle traveled about 500 m. Table III presents the MSE and the number of particles. Although UFastSLAM used one-fifth of the particles of FastSLAM2.0, the estimated error is almost the same.

TABLE III
SECOND EXPERIMENT

SLAM algorithm	Number of particles	MSE[m] (size of environment, traveling)
FastSLAM2.0	50	1.6815 (100m x 100m, $\simeq$ 500m)
UFastSLAM	10	1.6963 (100m x 100m, $\simeq$ 500m)

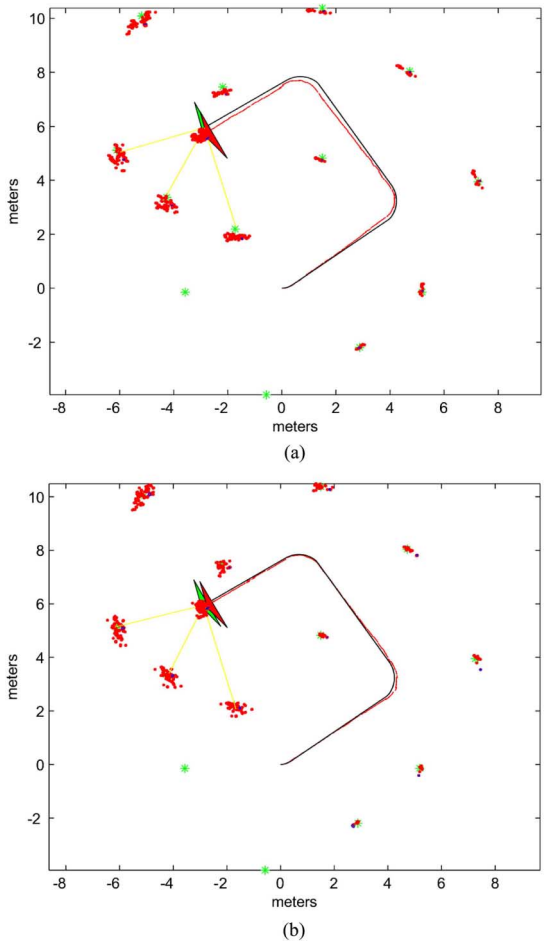


Fig. 5. (a) FastSLAM2.0 and (b) UFastSLAM. The black line and green (bright) asterisks denote the true and landmark positions, respectively. The red (bright) line is the mean estimate of the vehicle position and the red (bright) dots, near the asterisks, are estimated landmarks. These cases used 500 particles.

C. Evaluation of Filter Consistency

In this section, the consistency of both the UFastSLAM and the FastSLAM2.0 is considered with same resampling strategy.

To verify the consistency of both algorithms, average normalized estimation error squared (NEES) is used as a measure. Thirty Monte Carlo simulations were performed with the two-sided 95% probability concentration region. The acceptance interval of this condition is in [2.19, 3.93].

The vehicle used in this experiment had a 0.26 m wheelbase with 0.6 m/s velocity. The control and sensor frequencies were 40 and 5 Hz, respectively. The control noises were 0.3 m/s in velocity and 3° in steering angle, and the measurement noises were 0.1 m in range and 1° in bearing. The maximum sensor range was 5 m, with a 180° frontal field-of-view. The environment used in this experiment is shown in Fig. 5 with the estimate results of both algorithms.

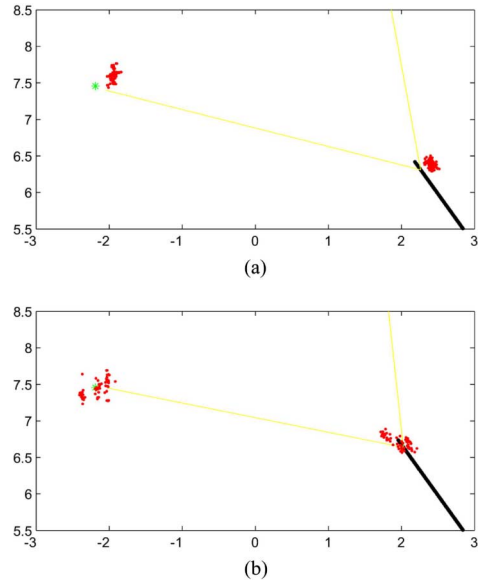


Fig. 6. (a) FastSLAM2.0 and (b) UFastSLAM. The black thick line and green (bright) asterisk denote the true path and landmark position, respectively. The red (bright) dots, near the true vehicle pose, are the estimated vehicle states and the red (bright) dots, near the green asterisk, are the estimated landmark location. Here, 100 particles were used for both algorithms.

Estimating an accurate state and its uncertainty is an important problem for improving the consistency of the SLAM filter. Consider the two situations depicted in Fig. 6. The particle diversity of both cases is similar, because both algorithms use the same resampling strategy. Actually, both algorithms resampled similar numbers of times. However, since an inaccurate state estimation induced a convergence of the particles to a wrong pose, the estimated covariance of Fig. 6(a) does not contain the true vehicle pose and the consistency of Fig. 6(a) is broken. Fig. 6(a) has the possibility of an optimistic estimate. This is a general case of FastSLAM2.0.

On the other hand, although the diversity of Fig. 6(b) is similar to that of Fig. 6(a), the mean and covariance of Fig. 6(b) are estimated well. As a result, the uncertainty of Fig. 6(b) contains the true vehicle state. Since a more accurate estimate without accumulating linearization error is achieved in UFastSLAM, many particles can have accurate information repetitively. This induces the possibility for maintaining the consistency. Fig. 7 shows that the consistency of UFastSLAM with 100 particles is prolonged longer than that of FastSLAM2.0 with 500 particles.

VI. Experimental Results

A. Outdoor SLAM Result With Laser Scanner

UFastSLAM was compared to FastSLAM2.0 using the Sydney Victoria Park dataset, a popular dataset in the SLAM community. Fig. 8(a) shows Victoria Park with GPS data that is represented by an intermittent yellow line. The vehicle had a 2.83 m wheel base and was equipped with the SICK laser range finder with a 180° frontal field-of-view. The vehicle was driven around the park for about 30 min, covering a path of over 3.5 km. The velocity and the steering angle were measured with

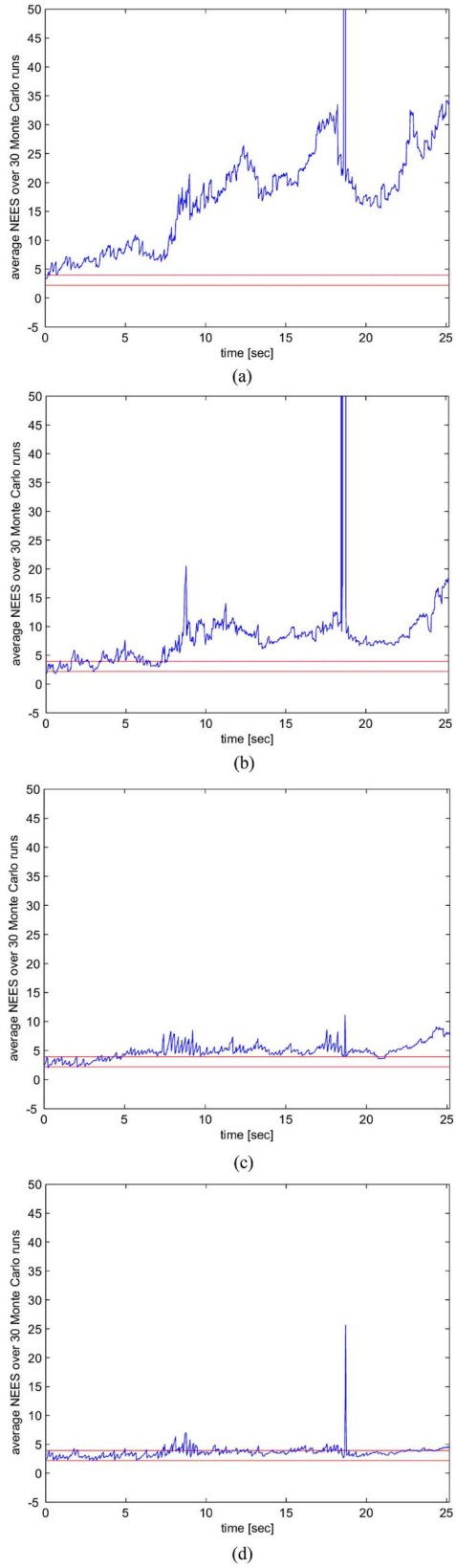
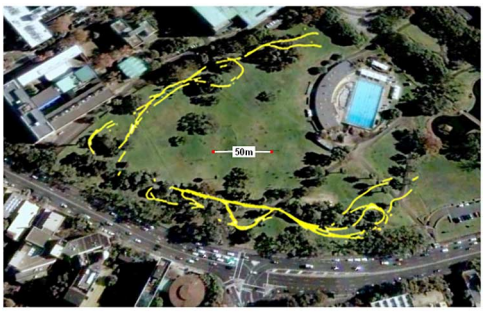
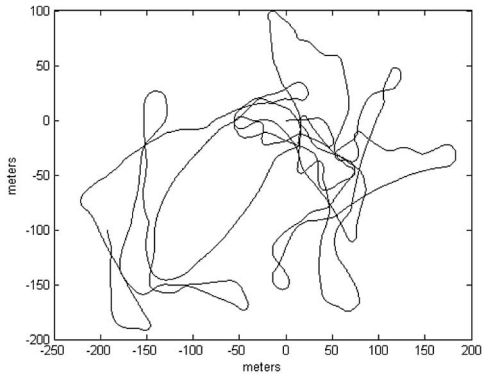


Fig. 7. (a) Consistency of FastSLAM2.0 with 100 particles. (b) Consistency of FastSLAM2.0 with 500 particles. (c) Consistency of UFastSLAM with 100 particles. (d) Consistency of UFastSLAM with 500 particles. The red bounds are the two-sided 95% probability concentration region with 30 Monte Carlo runs. The peaks at 18 s originated from large heading errors because of a rapid turn.



(a)

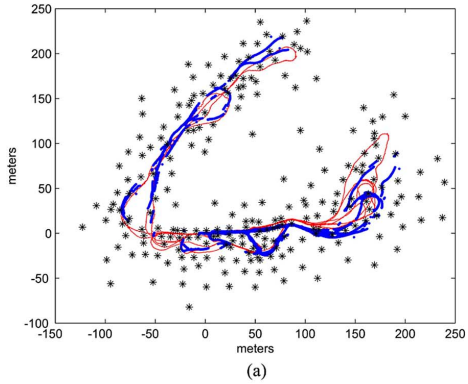


(b)

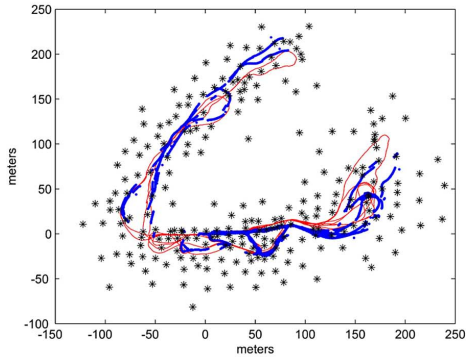
Fig. 8. (a) Victoria Park, Sydney, Australia, with the GPS path. The yellow (bright), intermittent path on the park is the GPS path of the vehicle. (b) Odometry of the vehicle.

encoders but uneven terrain induced additional nonsystematic errors because of wheel slippage and vehicle attitude. Consequently, the odometry information from the encoder is poor, as shown in Fig. 8(b). In the park, nearby trees were used as point landmarks, and there were many spurious observations including dynamic objects. Although the vehicle was equipped with the GPS, the sensor gave intermittent information due to limited satellite availability. Nevertheless, the ground true position of the vehicle from the GPS was good enough to validate the estimated vehicle state of the filter. In the experiment, ten particles were used for both algorithms. Each algorithm was run many times to confirm the variance of the estimate error, and the comparison results of the accuracy and the loop closure ability of each filter are shown in Fig. 9.

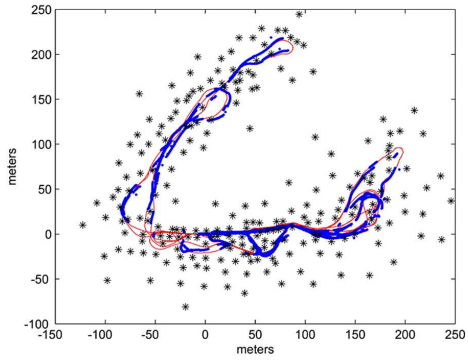
The results show that for both ten- and one-particle cases, the performance of UFastSLAM is better than that of FastSLAM2.0. The estimated paths of the UFastSLAM coincide very well with the GPS paths. Since the Kalman gains in the proposal distribution and the feature update are calculated from the unscented filter without using the Jacobian and the linearization of the nonlinear model, the uncertainties are propagated well and the accuracy of the state estimation has been improved over the FastSLAM approach. Furthermore, we observed from many runs of each algorithm that, similar to the simulation results, the UFastSLAM showed a smaller variance in the estimation error. In other words, the UFastSLAM almost always had the same estimate results, which are shown in Fig. 9(c) and (d). However, FastSLAM2.0 had a relatively large variance in the estimate results, with larger estimation errors than UFastSLAM.



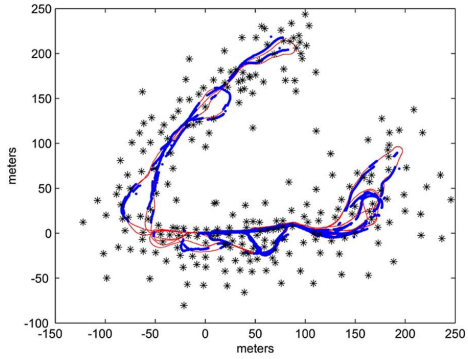
(a)



(b)



(c)



(d)

Fig. 9. (a) FastSLAM2.0 with ten particles. (b) FastSLAM2.0 with one particle. (c) UFastSLAM with ten particles. (d) UFastSLAM with one particle. In all figures, the thick blue path is the GPS data and the solid red path is the estimated path. The black asterisks are the estimated positions of the landmark. The control noises in the experiment are $\sigma_V = 0.8\ \mathrm{m/s}$, $\sigma_\gamma = 1.8°$, and the measurement noises are $\sigma_r = 1.5\ \mathrm{m}$, $\sigma_\phi = 2.8°$. For the unknown data association, the individual compatibility nearest neighbor test with 2σ acceptance region is used.

(a) (b)

Fig. 10. (a) Indoor environment for the experiment. Here, the robot is at (5.5 m, 0.5 m) in Fig. 11(b). (b) Our indoor mobile robot. In this experiment, the only sensor was a ring of 12 sonars.

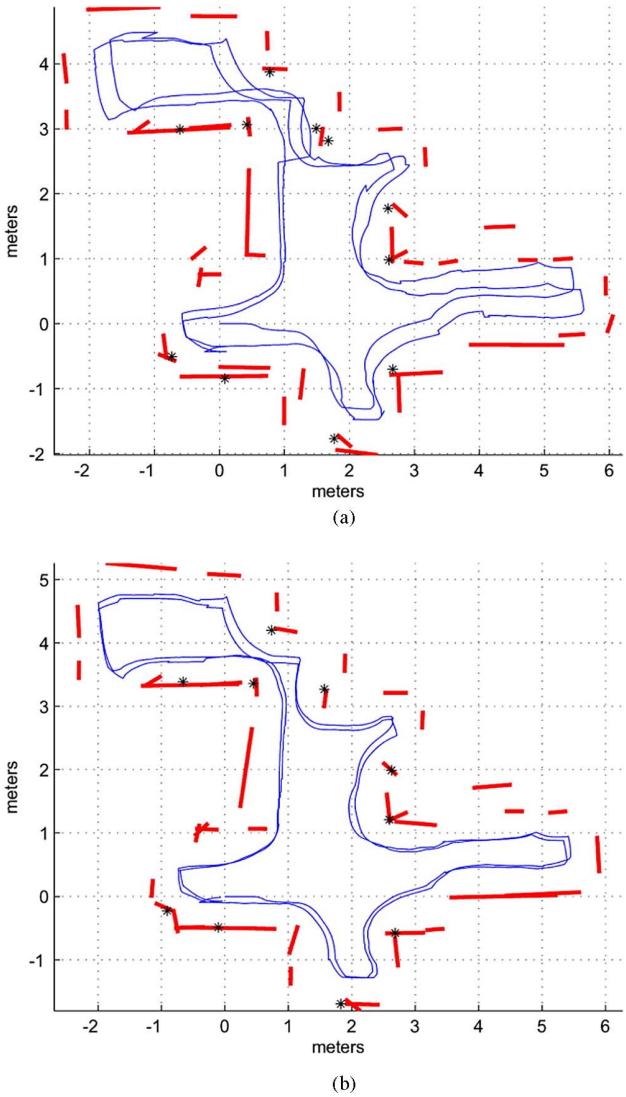


Fig. 11. Estimated maps and robot trajectories. (a) FastSLAM2.0. (b) UFastSLAM. The red thick lines and the black asterisks are the sonar line and the point features, respectively. The blue thin lines are the estimated robot trajectories.

B. Indoor SLAM Result With Sonar Sensors

To validate the robustness of UFastSLAM to high sensor noise, many experiments were carried out in indoor environments using sonar features (line and point [32]), one of which is described in this section. In the home environment shown in Fig. 10(a), the robot explored the environment following the right side obstacle, and after closing the loop, it traveled the environment using the same navigation strategy one more time. A sonar ring consisting of 12 sonars was attached to a ActiveMedia Pioneer 3 DX, as shown in Fig. 10(b) and a 2-GHz laptop was used.

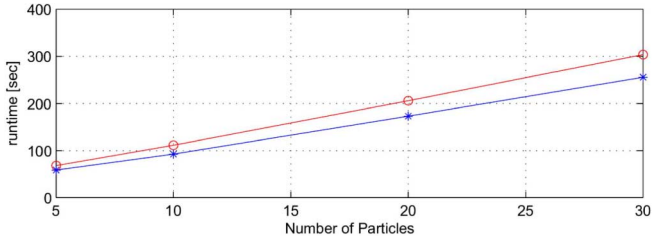


Fig. 12. Average runtime of the indoor experiment. The circles are UFastSLAM and the asterisks are FastSLAM2.0. In this experiment, the binary tree representation of the landmarks is not considered.

Fig. 11 shows the final estimated map and the trajectory of the robot. FastSLAM2.0 failed to close the loop and it generated an inconsistent map as shown in Fig. 11(a), since the uncertainty of sonar features is large (0.1–0.2 m in range, 5.5–6° in bearing). On the other hand, UFastSLAM estimated the robot trajectory and the map correctly, as shown in Fig. 11(b). In this experiment, 20 particles were used to represent the posterior.

To validate the complexity and the computational cost, we analyzed the execution time of the experiment. Fig. 12 describes the average execution time of both algorithms. The complexity of both algorithms is linear, since we did not use the binary tree technique for the landmark representation in this experiment. The computational cost of UFastSLAM is slightly larger than FastSLAM2.0.

VII. Conclusion

This paper proposed the UFastSLAM algorithm as a robust and effective SLAM solution. The main advantage of this algorithm is that it does not use the derivation of the Jacobian matrices and the linear approximations to the nonlinear functions in the RBPF framework.

The proposed algorithm updates the mean and the covariance of the feature state by using the unscented filter to avoid linearization errors and Jacobian calculations in the feature estimation. New features are also registered by using an accurate transformation, SUT. In the localization process, the SUT is implemented in the prediction step of the vehicle state, and the unscented filter provides better proposal distribution without accumulating linearization errors and without calculating the Jacobian matrices in the measurement update step.

Due to this, the accuracy of the state estimation has been improved over the previous approaches. This means that the proposed algorithm has the additional advantage of reducing the number of particles while maintaining the estimation accuracy.

From the consistency viewpoint, since the vehicle and the feature states are estimated without accumulating linearization errors in UFastSLAM, many particles can have accurate information repetitively. Thus, although the particle diversity is

similar to the previous algorithm, more accurate estimation can be achieved consistently for longer time periods.

We conclude from the results in large-scale environments with various sensor noise uncertainties that the proposed UFastSLAM algorithm, even with fewer particles, yields significantly more accurate and robust estimation results compared with the previous linearization approaches. Additionally, we conclude that the consistency of UFastSLAM, even with fewer particles, is prolonged longer than that of FastSLAM2.0.

However, since the UFastSLAM also employs the resampling process, the reduction of the impact of the resampling should be treated more thoroughly to avoid an underestimate to the uncertainty .

APPENDIX

A. Detailed Equations for Passing the Sigma Points Through the Motion Model

To simplify the exposition, we introduce the following notations. Each ith *sigma points* state component, $\chi_{t-1}^{[i][m]}$, consists of the three components

$$\chi_{t-1}^{[i][m]} = \begin{bmatrix} \chi_{x,t-1}^{[i][m]} \\ \chi_{y,t-1}^{[i][m]} \\ \chi_{\theta,t-1}^{[i][m]} \end{bmatrix}. \tag{27}$$

Let us suppose the Ackerman model. The control inputs $[V_t \ G_t]^T$, with the added control noise component $\chi_t^{u[i][m]}$, are as follows:

$$\begin{bmatrix} V_n \\ G_n \end{bmatrix} = \begin{bmatrix} V_t + \chi_{V,t}^{u[i][m]} \\ G_t + \chi_{G,t}^{u[i][m]} \end{bmatrix} \tag{28}$$

where

$$\chi_t^{u[i][m]} = \begin{bmatrix} \chi_{V,t}^{u[i][m]} \\ \chi_{G,t}^{u[i][m]} \end{bmatrix}. \tag{29}$$

Then, the state components $\chi_{t-1}^{[i][m]}$ of the sigma points are passed through the Ackerman model:

$$\bar{\chi}_t^{[i][m]} = \chi_{t-1}^{[i][m]} + \begin{bmatrix} V_n \ \Delta t \ \cos\left(G_n + \chi_{\theta,t-1}^{[i][m]}\right) \\ V_n \ \Delta t \ \sin\left(G_n + \chi_{\theta,t-1}^{[i][m]}\right) \\ V_n \, \Delta t \ (\sin(G_n)/\text{wheelbase}) \end{bmatrix}. \tag{30}$$

B. Augmented Approach for the Feature Update

For the feature update, the augmented state of the feature is constructed using the previously registered mean and covariance of the feature [24]:

$$\mu_{n_t,t-1}^{a[m]} = \begin{bmatrix} \mu_{n_t,t-1}^{[m]} \\ \mathbf{0} \end{bmatrix}, \quad \Sigma_{n_t,t-1}^{a[m]} = \begin{bmatrix} \Sigma_{n_t,t-1}^{[m]} & \mathbf{0} \\ \mathbf{0} & R_t \end{bmatrix} \tag{31}$$

where $\mu_{n_t,t-1}^{[m]}$ is the mean of the nth feature that is registered in feature initialization step. $\Sigma_{n_t,t-1}^{[m]}$ is the covariance matrix of the nth feature, and it has 2×2 dimensions in the RBPF framework. R_t is the measurement noise covariance.

The *sigma points* are defined using these augmented states, and the Kalman gain $\bar{K}_t^{[m]}$ is calculated by (19) without using R_t as an additive term in the innovation covariance $\bar{S}_t^{[m]}$.

ACKNOWLEDGMENT

The authors would like to thank T. Bailey for providing the FastSLAM code, K. Lee for supporting the navigation program, and the University of Sydney for the use of the Victoria Park dataset.

REFERENCES

[1] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: Part I," *IEEE Robot. Autom. Mag.*, vol. 13, no. 2, pp. 99–108, Jun. 2006.

[2] T. Bailey and H. Durrant-Whyte, "Simultaneous localization and mapping: Part II," *IEEE Robot. Autom. Mag.*, vol. 13, no. 3, pp. 108–117, Sep. 2006.

[3] R. Martinez-Cantin and J. A. Castellanos, "Unscented SLAM for large-scale outdoor environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2005, pp. 328–333.

[4] S. Huang and G. Dissanayake, "Convergence and consistency analysis for extended Kalman filter based SLAM," *IEEE Trans. Robot.*, vol. 23, no. 5, pp. 1036–1049, Oct. 2007.

[5] R. Martinez-Cantin and J. A. Castellanos, "Bounding uncertainty in EKF-SLAM: The robocentric local approach," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2006, pp. 430–435.

[6] J. A. Castellanos, J. Neira, and J. D. Tardos, "Limits to the consistency of EKF-based SLAM," in *Proc. 5th IFAC Symp. Intell. Auton. Vehicles*, Lisbon, Portugal, Jul. 2004, pp. 1244–1249.

[7] T. Bailey, J. Nieto, J. Guivant, M. Stevens, and E. Nebot, "Consistency of the EKF-SLAM algorithm," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2006, pp. 3562–3568.

[8] T. Bailey, J. Nieto, and E. Nebot, "Consistency of the FastSLAM algorithm," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2006, pp. 424–429.

[9] J. A. Castellanos, R. Martinez-Cantin, J. D. Tardos, and J. Neira, "Robocentric map joining: Improving the consistency of EKF-SLAM," *Robot. Auton. Syst.*, vol. 55, pp. 21–29, 2007.

[10] Y. Bar Sahlom, X. R. Li, and T. Kirubarajan, *Estimation With Applications to Tracking and Navigation*. New York: Wiley, 2001.

[11] J. D. Tardos, J. Neira, P. M. Newman, and J. J. Leonard, "Robust mapping and localization in indoor environments using sonar data," *Int. J. Robot. Res.*, vol. 21, no. 4, pp. 311–330, 2002.

[12] C. Estrada, J. Neira, and J. D. Tardos, "Hierarchical SLAM: Real-time accurate mapping of large environments," *IEEE Trans. Robot.*, vol. 21, no. 4, pp. 588–596, Aug. 2005.

[13] S. Julier, J. Uhlmann, and H. F. Durrant-Whyte, "A new method for the nonlinear transformation of means and covariances in filters and estimators," *IEEE Trans. Autom. Control*, vol. 45, no. 3, pp. 477–482, Mar. 2000.

[14] S. Julier and J. K. Uhlmann, "A counter example to the theory of simultaneous localization and map building," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2001, pp. 4238–4243.

[15] S. Julier and J. K. Uhlmann, "Unscented filtering and nonlinear estimation," *Proc. IEEE*, vol. 92, no. 3, pp. 401–422, Mar. 2004.

[16] S. Julier, "The scaled unscented transformation," in *Proc. Amer. Control Conf.*, 2002, pp. 4555–4559.

[17] R. Merwe, A. Doucet, N. Freitas, and E. Wan, "The unscented particle filter," Cambridge Univ., Eng. Dep., Cambridge, U.K., Tech. Rep. CUED/F-INFENG/TR 380. 2000.

[18] K. Murphy, "Bayesian map learning in dynamic environments," in *Neural Information Proceedings System (NIPS)*. Cambridge, MA: MIT Press, 1999.

[19] R. Martinez-Cantin, N. de Freitas, and J. A. Castellanos, "Analysis of particle methods for simultaneous robot localization and mapping and a new algorithm: Marginal-SLAM," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2007, pp. 2415–2420.

[20] K. R. Beevers and W. H. Huang, "Fixed-lag sampling strategies for particle filtering SLAM," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2007, pp. 2433–2438.

[21] M. Montemerlo and S. Thrun, "Simultaneous localization and mapping with unknown data association using FastSLAM," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2003, pp. 1985–1991.
[22] M. Montemerlo, "FastSLAM: A factored solution to the simultaneous localization and mapping problem with unknown data association," Ph.D. dissertation, Carnegie Mellon Univ., Pittsburgh, PA, 2003,
[23] C. Kim, R. Sakthivel, and W. K. Chung, "Unscented FastSLAM: A robust algorithm for the simultaneous localization and mapping problem," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2007, pp. 2439–2445.
[24] X. Wang and H. Zhang, "A UPF-UKF framework for SLAM," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2007, pp. 1664–1669.
[25] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. Cambridge, MA: MIT Press, 2005.
[26] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with Rao–Blackwellized particle filters," *IEEE Trans. Robot.*, vol. 23, no. 1, pp. 34–46, Feb. 2007.
[27] P. Beeson, A. Murarka, and B. Kuipers, "Adapting proposal distributions for accurate, efficient mobile robot localization," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2006, pp. 49–55.
[28] T. D. Barfoot, "Online visual motion estimation using FastSLAM with SIFT features," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syt.*, 2005, pp. 579–585.
[29] R. Merwe, "Sigma-point Kalman filters for probabilistic inference in dynamic state-space models," Ph.D. dissertation, OGI School of Sci. & Eng., Oregon Health and Sci. Univ., Portland, OR, Apr. 2004.
[30] A. Doucet, N. de Freitas, and N. Gordan, *Sequential Monte Carlo Methods in Practice*. New York: Springer-Verlag, 2001.
[31] (2008, Jun.). [Online]. Available: http://www-personal.acfr.usyd.edu.au/tbailey/software/index.html
[32] J. Choi, S. Ahn, M. Choi, and W. K. Chung, "Metric SLAM in home environment with visual objects and sonar features," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syt.*, 2006, pp. 4048–4053.

Chanki Kim (S'07) received the B.S. degree in mechanical engineering from Hanyang University, Seoul, Korea, in 2003, and the M.S. degree in mechanical engineering, in 2005 from Pohang University of Science and Technology (POSTECH), Pohang, Korea, where, he is currently working toward the Ph.D. degree.

His current research interests include mobile robot simultaneous localization and mapping (SLAM), path planning, and integrated exploration.

Rathinasamy Sakthivel received the B.Sc., M.Sc., and Ph.D. degrees in mathematics from Bharathiar University, Coimbatore, India, in 1992, 1994, and 1999, respectively.

He is currently a Postdoctoral Researcher in the Department of Mechanical Engineering, Pohang University of Science and Technology (POSTECH), Pohang, Korea. From 2003 to 2005, he was a JSPS (Japan Society for the Promotion of Science) Fellow in the Department of Systems Innovation and Informatics, Kyushu Institute of Technology, Japan. His current research interests include autonomous mobile robots, simultaneous localization and mapping (SLAM) problems, and robust control for nonlinear systems.

Wan Kyun Chung (M'84) received the B.S. degree in mechanical design from Seoul National University, Seoul, Korea, in 1981, the M.S. degree in mechanical engineering and the Ph.D. degree in production engineering from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 1983, 1987, respectively.

In 1987, he was a Professor in the School of Mechanical Engineering, Pohang University of Science and Technology (POSTECH), Pohang, Korea. In 1988, he was a Visiting Professor at the Robotics Institute Carnegie-Mellon University, Pittsburgh, PA. In 1995, he was a Visiting Scholar at the University of California, Berkeley. He is currently working as the Director of the National Research Laboratory for Intelligent Mobile Robot Navigation. He is also on the International Editorial Board for *Advanced Robotics*. His current research interests include localization and navigation of mobile robots, underwater robots, and development of robust controllers for precision motion control.

Dr. Chung is an Associate Editor for IEEE TRANSACTION ON ROBOTICS.