VOL. 11, NO. 10, MAY 2016 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences


www.arpnjournals.com

SIGMA POINT-BASED FASTSLAM: SOLUTION TO SLAM PROBLEM

Z. Zafizal, R. Saifudin, D. Mohd Razali and A. Hamzah
Telekom Malaysia, [2]Faculty of Electrical and Electronics Engineering, Universiti Malaysia Pahang, Malaysia
E-Mail: zafizal@tm.com.my

ABSTRACT

This paper proposes a reduced sigma point transformation for a FastSLAM framework. The sigma point transformation is used to estimate robot poses in conjunction with generic particle filter used in standard FastSLAM framework. This method can estimate robot poses more consistently and accurately than the current standard particle filters, especially when involving highly nonlinear models or non-Gaussian noises. In addition, this algorithm avoids the calculation of the Jacobian for motion model which could be extremely difficult for high order systems. We proposed a sampling strategy known as a spherical simplex for sigma point transformation to estimate robot poses in FastSLAM framework. Simulation results are shown to validate the performance goals.

Keywords: simultaneous localization and Mapping, extended kalman filter, unscented kalman filter, FastSLAM, particle filter, spherical simplex unscented transformation.

INTRODUCTION

In recent years, a research on SLAM were introduced by Smith and Cheesman [1], in which the use of the extended Kalman filter (EKF) was proposed for solving SLAM and it was firstly implemented by [2]. However, there are two limitations of the EKF approach. One is the high computational cost to maintain a multivariate Gaussian vector which requires quadratic time in the dimension of the map. Therefore, it was proposed in [3] and [4] to build a set of smaller maps and then combined them to build a large one. The other is related to the data association problem. It is critical to choose the correct data association hypotheses because different data association hypotheses lead to different maps. Maintaining posteriors over multiple data associations makes the SLAM algorithms more robust. Unfortunately, Gaussian distributions cannot represent multi-modal ones, so only the most likely data association can be incorporated. As a result, the approach tends to fail catastrophically when the incorporated data association is incorrect.

Alternative family of SLAM algorithms is called FastSLAM [5]. It was pointed out in [6] that the errors of the feature estimates would be independent if a robot path was given. This property is a base of FastSLAM algorithms to solve the SLAM problems. Particle filters are used to estimate the robot path. Conditioned on these particles, the mapping problem is factored into separate problems. Therefore, one EKF for each feature is used to update the feature estimate. However, the standard FastSLAM frameworks used an EKF to improve the accuracy of a proposal distribution, but the EKF involves Jacobian matrices and the linear approximations of the nonlinear functions. Calculating the Jacobian is uninvited effort, and inaccurate approximation to the posterior covariance deteriorates the estimate accuracy and the filter consistency. Therefore, in this paper we introduce a sigma point transformation in the FastSLAM framework to eliminate the linearization as well as the Jacobian calculation. In Kim *et al.* [7], the authors proposed a sigma point transformation for all components of FastSLAM framework. This will increase total computational cost especially in calculation number or sigma points for each component of FastSLAM known as robot pose estimation, landmark estimation and weight calculation. In our proposed method, we are focusing the sigma point transformation for robot pose estimation only and the others remain unchanged. In addition, we also proposed a different sampling strategy known as a spherical simplex to unscented transformation to estimate robot poses in FastSLAM framework. We then compared conventional types of standard sigma point FastSLAM with proposed sampling method known as spherical simplex sigma point FastSLAM, respectively.

The structure of this paper is as follows. In next section, we generally describe about the FastSLAM framework follows by presenting proposed sampling strategy of the sigma point transformation for the FastSLAM. A simulation setup for a SLAM problem is shown in next following section, as well as discussion on the simulation results. The paper is concluded in last section.

SIGMA POINT FastSLAM

In general, the FastSLAM2.0 framework consists of three parts: the robot state estimation, the feature state estimation, and the importance weight calculation as deeply discussed in [8, 9]. In this section, the sigma point transformation to the robot state estimation is discussed in detail. The last two parts still remain unchanged as applying to FastSLAM2.0 and will be not covered in this paper. We proposed two different sampling strategies to sigma point transformation known as symmetrical sigma point and spherical simplex sigma point and they are discussed in detail in the following subsection. In our mind, the robot state estimation is crucial to develop accurate maps as well as to reduce overall computational cost. In addition, this project is an extension of previous work that can be referred to in [10].

Symmetrical sigma point FastSLAM framework

The FastSLAM uses a particle filter to sample over robot paths and each particle processes N low-dimensional EKFs, once for each of the N landmarks. Instead of using linearizing through the 1st-order Taylor series expansion used by EKF, the proposed method introduces a set of deterministic points known as sigma points to propagate them through nonlinear model [11]. In other word, we replace the EKF used in the standard FastSLAM by sigma point transformation method. To avoid confusion, the use of a particle is referring to the component used by particle filter to sample robot paths, while a sigma point is referring to the component used by an unscented transformation to replace the work performed by the EKF in standard FastSLAM framework.

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 the other is the measurement update step. At first, the state vector is augmented with a control input and the observation.

$$x_{t-1}^{a[m]} = \begin{bmatrix} x_{t-1}^{[m]} \\ 0 \\ 0 \end{bmatrix}, \qquad P_{t-1}^{a[m]} = \begin{bmatrix} P_{t-1}^{[m]} & 0 & 0 \\ 0 & Q_t & 0 \\ 0 & 0 & R_t \end{bmatrix}$$

Here, $x_{t-1}^{a[m]}$ and $P_{t-1}^{a[m]}$ are the augmented vector for the state and related covariance, respectively. $x_{t-1}^{[m]}$ is the previous mean of the robot as well as it covariance, $P_{t-1}^{[m]}$. Q_t and R_t are the control noise covariance and the measurement noise covariance, respectively.

A symmetric set of $2L+1$ sigma points $\chi_{t-1}^{a[i][m]}$ for the augmented state vector can be calculated as

$$\chi_{t-1}^{a[i][m]} = \left[x_{t-1}^{a[i][m]} \quad \cdots \quad x_{t-1}^{a[i][m]}\right] + \gamma \left[0 \quad \sqrt{P_{t-1}^{a[i][m]}} \quad -\sqrt{P_{t-1}^{a[i][m]}}\right] \tag{1}$$

where $\gamma = \sqrt{L+\lambda}$ and L is the augmented state vector dimension. The $\lambda = \alpha^2(L+\kappa) - L$ and α value between zero and one should be a small number to avoid sampling nonlocal effects when the nonlinearities are strong. κ is a scaling parameter that determines how far the sigma points are separated from the mean and a good default choice is $\kappa = 0$ [14].

The set of sigma points $\chi_{t-1}^{a[i][m]}$ is then propagated through the motion model $\boldsymbol{h}(\cdot)$ given by

$$\bar{\chi}_{t-1}^{[i][m]} = \boldsymbol{h}\left(\chi_{t-1}^{[i][m]}, \chi_t^{u[i][m]}\right) \tag{2}$$

Here, $\bar{\chi}_{t-1}^{[i][m]}$ is the transformed sigma points of the robot state. $\chi_{t-1}^{[i][m]}$ and $\chi_t^{u[i][m]}$ are the part of augmented sigma points related to the robot poses and control component, respectively.

The first two moments of the predicted robot state are computed by the following equations:

$$\hat{x}_{t-1}^{[m]} = \sum_{i=0}^{2L} \omega_m^i \, \bar{\chi}_t^{[i][m]} \tag{3}$$

$$\hat{P}_{t-1}^{[m]} = \sum_{i=0}^{2L} \omega_c^i \left(\bar{\chi}_t^{[i][m]} - \hat{x}_{t-1}^{[i][m]}\right)\left(\bar{\chi}_t^{[i][m]} - \hat{x}_{t-1}^{[i][m]}\right)^T \tag{4}$$

where the constant weights ω_m^i and ω_c^i are parameters related to computing mean and covariance given as follows in [14]. As some features are observed, the data association provides their identities, and the updating step can be performed. The measurement sigma points $\bar{z}_t^{[i][m]}$ are calculated using the observation model $\boldsymbol{g}(\cdot)$, function of sigma points of robot poses $\bar{\chi}_t^{[i][m]}$ and measurement noise $\chi_t^{z[i][m]}$ given below:

$$\bar{Z}_t^{[i][m]} = \boldsymbol{g}\left(\bar{\chi}_{t-1}^{[i][m]}, \chi_t^{z[i][m]}\right) \tag{5}$$

Then, the measurement mean $\hat{\mu}_{Z_t}^{[m]}$, covariance $\sum_{Z_t}^{[m]}$, and cross covariance $\sum_{x_t Z_t}^{[m]}$ as well as the Kalman gain $\boldsymbol{K}_t^{[m]}$ can be computed as follows:

$$\hat{\mu}_{Z_t}^{[m]} = \sum_{i=0}^{2L} \omega_m^i \, \bar{Z}_t^{[i][m]} \tag{6}$$

$$\sum_{Z_t}^{[m]} = \sum_{i=0}^{2L} \omega_c^i \left(\bar{Z}_t^{[i][m]} - \hat{\mu}_{Z_t}^{[m]}\right)\left(\bar{Z}_t^{[i][m]} - \hat{\mu}_{Z_t}^{[m]}\right)^T \tag{7}$$

$$\sum_{x_t Z_t}^{[m]} = \sum_{i=0}^{2L} \omega_c^i \left(\bar{\chi}_t^{[i][m]} - \hat{x}_{t-1}^{[i][m]}\right)\left(\bar{Z}_t^{[i][m]} - \hat{\mu}_{Z_t}^{[m]}\right)^T \tag{8}$$

$$K_t^{[m]} = \sum_{x_t Z_t}^{[m]} \left(\sum_{x_t Z_t}^{[m]}\right)^{-1} \tag{9}$$

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

$$\mu_{x_t}^{[m]} = \hat{x}_{t-1}^{[m]} + K_t^{[m]} \left(z_t - \hat{\mu}_{Z_t}^{[m]}\right) \tag{10}$$

$$\sum_{x_t}^{[m]} = P_{t-1}^{[m]} - K_t^{[m]} \sum_{Z_t}^{[m]} \left(K_t^{[m]}\right)^T \tag{11}$$

However the computational cost highly depended on the number of sigma points used, as shown in Equation. (1). In order to reduce this number of sigma points used, we propose the spherical simplex sigma point transformation sampling technique in the following section.

Spherical simplex sigma point transformation approach

The selection criterion of spherical simplex sigma points is a new and better sampling strategy for the unscented transformation [12-13], which significantly allows reducing the number of sigma points propagated. By this fact, the implementation of filters becomes more suitable for real-time systems, where the limitations of computational cost are extremely restrictive.

This sampling strategy defines a minimum set of points located in a hyper-sphere. For an L-dimensional space, only $L + 2$ sigma points are required. These sigma points are in a radius that is proportional to $\sqrt{L}$, and the weight applied to each point is proportional to $1/L$.

In a case of a FastSLAM, a set of $L + 2$ sigma points $\chi_{t-1}^{a[i][m]}$ for the augmented state vector can be calculated as

$$\chi_{t-1}^{a[i][m]} = x_{t-1}^{a[m]} + \sqrt{P_{t-1}^{a[m]}} Z_i \quad , i = 0, \dots, L+1 \tag{12}$$

Where $\sqrt{P_{t-1}^{a[m]}}$ indicates the square root of the covariance matrix $P_{t-1}^{a[m]}$, and Z_i is the ith column of the spherical simplex sigma point matrix. Z_i can be computed by the following algorithm:

- Choose the value $0 \leq W_0 \leq 1$.
- The sequence of weights is chosen as $W_i = \frac{1-W_o}{L+1}$
- In order to use the advantages of the scaled transformation, the previous weights are transformed by the following way [16]:

$$\omega_t = \begin{cases} 1 + \frac{W_o}{\alpha^2} & for \quad i = 0 \\ \frac{W_i}{\alpha^2} & for \quad i \neq 0 \end{cases}$$

where α is the sigma points scalar factor ($0 < \alpha \leq 1$), which allows to minimize the higher order errors.

- The vector sequence is initialized as

$$Z_0^1 = [0], \; Z_1^1 = \left[\frac{1}{\sqrt{2\omega_1}}\right], \; Z_2^1 = \left[\frac{1}{\sqrt{2\omega_1}}\right]$$

- The vector sequence is expanded for $j = 2, \dots, L$ according to

$$Z_i^j = \begin{cases} \begin{bmatrix} Z_0^{j-1} \\ 0 \end{bmatrix} & i = 0 \\ \begin{bmatrix} Z_i^{j-1} \\ -\frac{1}{\sqrt{j(j+1)\omega_1}} \end{bmatrix} & i = 1, \dots, j \\ \begin{bmatrix} 0 \\ \frac{1}{\sqrt{j(j+1)\omega_1}} \end{bmatrix} & i = j+1 \end{cases}$$

- Finally, in order to incorporate information of higher order, define in [14]:

$$\omega_m^0 = \omega_0 \;, \; \omega_c^0 = \omega_0 + (1 - \alpha^2 + \beta) \; for \; i = 0,$$
$$\omega_m^i = \omega_0^i = \omega_i \quad for \; i = 1, \dots, j+1$$

where β denotes a parameter that affects the weight of the zeroth sigma point for the calculation of the covariance, allowing to minimize higher order errors if a previous knowledge of the distribution is provided for $\boldsymbol{x}$. Once the sigma points have been calculated such as in Equation. (12), the predicting and updating steps can be accomplished using standard unscented transformation as mentioned in previous subsection. This can be computed from Equation. (2) to Equation. (11).

SIMULATION

In order to verify the performance of the proposed framework, some experiments were conducted using simulation data with 20 feature landmarks and known data association. The simulator was developed based on the work in [14]. The exploration area of the vehicle is 40 meters wide and 100 meters long, and the landmarks are randomly located in the area.

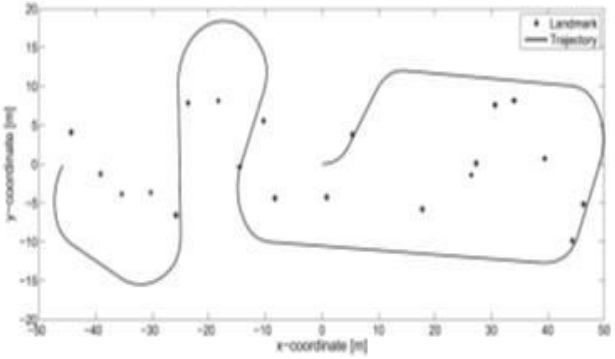


Figure-1. True trajectory and landmark.

Figure-1 shows the landmarks and true vehicle trajectory setup for this simulation. The vehicle starts at the initial pose $(0\,m, 0\,m, 0^o)$ and travels with a nominal speed and a steering angle of $3\,m/s$ and 30^o, respectively. The nominal control values are corrupted with Gaussian noises with standard deviations $0.3\,m/s$ and 3^o, respectively for each $0.025\,s$ sampling interval. The sensor takes measurements for each $0.2\,s$ time interval and the nominal measurement values are also assumed to be corrupted with Gaussian noises with standard deviations $0.1\,m$ and 1^0, respectively. In this simulation, we computed and compared a symmetrical sigma point FastSLAM (SSPFastSLAM) and a spherical simplex sigma point FastSLAM (SSSPFastSLAM). As initial conditions, we set the number of particles equal to 100 for both of these estimators.

Motion and observation model

In our simulation, we consider a simple Ackerman steered vehicle model as shown in Figure. 2. This vehicle is equipped with a set of dead-reckoning sensors that measure the back wheel speed and the steering angle. The nomenclatures are as follows:

$\Sigma w (O - YX)$	: absolute coordinate system
$\Sigma v (O_v - Y_v X_v)$	: vehicle coordinate system
x, y and ϕ	: position and orientation of the vehicle in Σw.
l	: a distance between front and rear wheel axles of the vehicle
V	: velocity of the vehicle
φ and θ	: steering angle of the vehicle and bearing of landmark in Σv

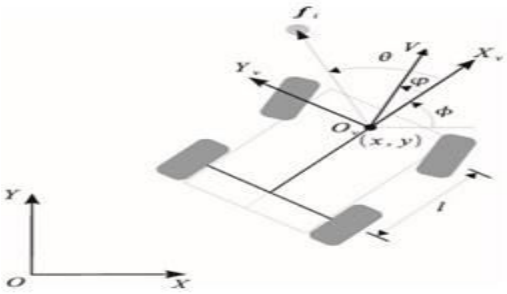


Figure-2. Vehicle in process of observing a feature.

The motion model of the vehicle is assumed to be described as follows:

$$\begin{bmatrix} x_{t+1} \\ y_{t+1} \\ \Phi_{t+1} \end{bmatrix} = \begin{bmatrix} x_t + V \cdot dt \cdot \cos(\varphi + \Phi_t) \\ y_t + V \cdot dt \cdot \sin(\varphi + \Phi_t) \\ \Phi_t + \dfrac{V \cdot dt}{t} \cdot \sin(\varphi) \end{bmatrix}$$

This vehicle is equipped with a range and bearing sensor. It can sense an object bounding in ± 30 degree semi-circle with the maximum range of 30 meter. The measurement equation is as follows:

$$\boldsymbol{Z}_t = \begin{bmatrix} \sqrt{(f_{i,x} - x_t)^2 + (f_{i,y} - y_t)^2} \\ \tan^{-1} \dfrac{f_{i,y} - y_t}{f_{i,x} - x_t} - \theta_t \end{bmatrix} + \mathbf{w}_t$$

where $(f_{i,x}, f_{i,y})$ is a landmark feature available in Cartesian coordinate system at time when the sensor takes a measurement. $\boldsymbol{w}_t \sim N(0, \boldsymbol{R}_t)$ is a zero-mean Gaussian white measurement noise with $\boldsymbol{R}_t = diag(0.1^2, 1^2)$ in unit $[m^2, deg^2]$.

Experimental results

Figure-3 shows the results of a SSPFastSLAM and a SSSPFastSLAM compared to true vehicle trajectory, respectively. Every estimator has been similarly performed at the beginning of the trajectory. However, their performances become different when the vehicle enters a first corner.

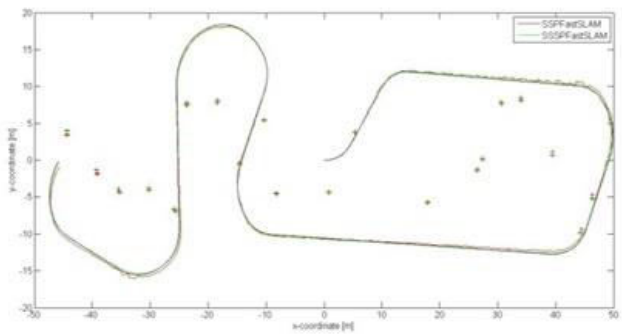
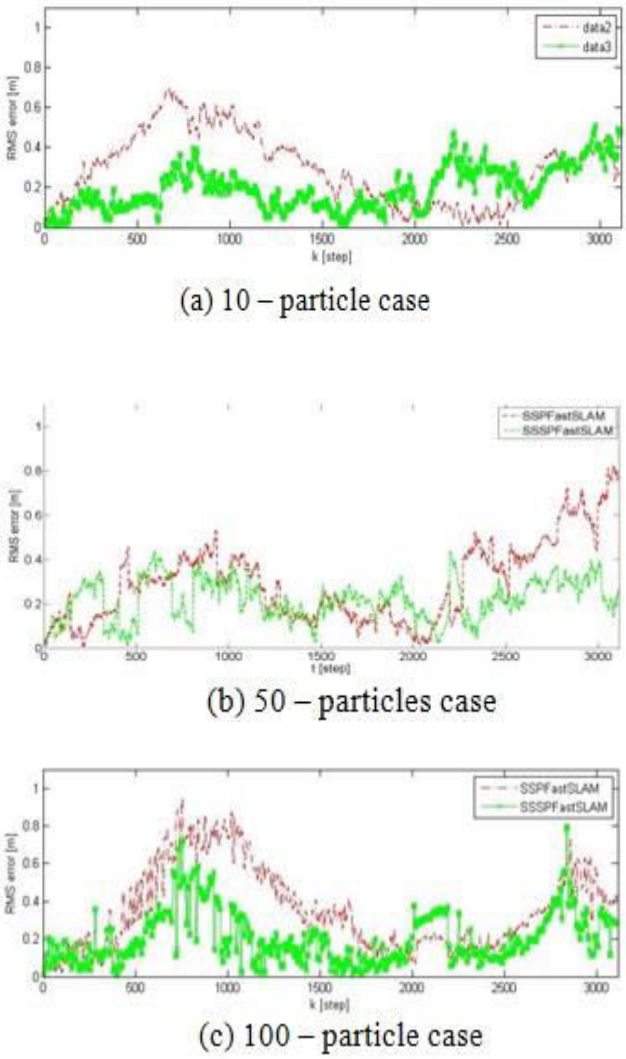


Figure-3. Estimates by symmetry sigma point FastSLAM, (SSPFastSLAM) and a spherical simplex sigma point FastSLAM (SSSPFastSLAM).

We cannot see a significant difference among both estimation methods because the axis scales used are quite large but in some place, we can clearly see that the proposed estimators drawn with a green-line respectively performed better than the standard symmetry sigma point FastSLAM. To obtain more promising result, we have computed the rms vehicle position error using the following equation,

$$RMS = \sqrt{\frac{1}{k}\sum_{i=1}^{k} (\hat{x}_i - x_i)^2 + (\hat{y}_i - y_i)^2}$$

where $\hat{x}_i$ and $\hat{y}_i$ are estimates for the vehicle position.



(a) 10 – particle case

(b) 50 – particles case

(c) 100 – particle case

Figure-4. RMS error for vehicle pose estimates by SSPFastSLAM, and SSSPFastSLAM with 10, 50 and 100-particles cases, respectively.

Furthermore, we investigate the effect of the number of particles used in estimation, we have reduced this particle number from 100 particles to 50 particle and

finally to 10 particles. Other simulation setup remains unchanged as previous one. We then calculated the root mean square (rms) error of x-axis, y-axis and vehicle position each filtering estimation method and with different number of particles as shown in Figure-4.

For overall rms errors, the proposed method, although the number of particles was reduced, remain at better performance compared to the standard symmetrical sigma point FastSLAM. It can be shows in Tables-1 which give a solid proved that the proposed method had better performance over the original sigma point FastSLAM.

We further analyze the impact of the number of sigma points used by the standard sigma point FastSLAM and the spherical simplex sigma point FastSLAM for processing time. By using build-in MATLAB function known as profiler and PC with 2.GHz CPU and 3.0GB RAM, processing time for both methods for the case of 10-, 50- and 100-particle has been calculated as shown in Table-2. From the table, it is clearly found that the required processing time depends on the total particle and also the number of sigma points used. As expected, the time required by the SSPFastSLAM was longer than the time spent by the SSSPFastSLAM. This is due to the fact that the SSSPFastSLAM uses only $n + 2$ sigma points while the SSPFastSLAM uses more sigma points ($2n + 1$). In general, although both methods have nearly the same performance in estimation, the SSSPFastSLAM has the advantage of fast in processing time.

Table-1. RMS errors with difference number of particle particle Error item Numb. of particl SSPFastSLAM SSSPFastSLAM.

Error item	Numb. of particle	SSPFastSLAM	SSSPFastSLAM
x-axis	10	0.108	0.112
	50	0.118	0.111
	100	0.104	0.113
y-axis	10	0.236	0.201
	50	0.185	0.132
	100	0.177	0.149
Pose	10	0.275	0.251
	50	0.238	0.186
	100	0.225	0.209

Table-2. Processing time for both FastSLAM. Numb. of particle SSPFastSLAM SSSPFastSLAM.

Numb. of particle	SSPFastSLAM	SSSPFastSLAM
10	34.92	31.13
50	160.40	149.63
100	326.55	288.11

CONCLUSIONS

In this paper, sigma point transformation for robot pose estimation has been applied to a FastSLAM framework and its performance has been also evaluated. The standard symmetrical sampling technique and a spherical simplex sampling technique used in sigma point transformation were evaluated. The simulation showed that the proposed methods gave better estimation, compared to the previous standard sigma point FastSLAM. Furthermore, we reduced the number of particles from 100 particles to 50 and 10 particles. As a result, the proposed methods still maintained its better performance compared to standard FastSLAM. The rms errors of x-axis, y-axis, and robot position then also clearly proved that the proposed methods had better performance than the standard FastSLAM.

ACKNOWLEDGEMENTS

This work was supported by the Ministry of Higher Education (Malaysia) and Universiti Malaysia Pahang under grant no. RDU130140.

REFERENCES

[1] Smith C. R. and Cheeseman. P. 1986. On the representation and estimation of spatial uncertainty. Int. Journal Robotics Research. Vol. 5(4). pp. 56–68.

[2] Moutarlier P. and Chatila R. 1989. Stochastic multisensory data fusion for mobile robot location and environment modeling. 5th Int. Symposium on Robotics Research, Tokyo. pp. 207–216.

[3] Thrun S., Koller D., Ghahramani, D., Durrant-Whyte H., and Ng, A.Y. 2004. Simultaneous mapping and localization with sparse extended information filters. Int. Journal of Robotics Research, vol. 23(7-8). pp. 693–716.

[4] Leonard J. J. and Feder H. J. S. 1999. A computationally efficient method for large-scale concurrent mapping and localization. J. Hollerbach and D. Koditschek, editors, Proc. Of the Ninth International Symp. on Robotics Research, Salt Lake City, Utah. pp. 169–176.

[5] Montemerlo M. and Thrun S. 2003. Simultaneous localization and mapping with unknown data association using FastSLAM, Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA). Vol. 2. pp. 1985–1991.

[6] Murphy K. 1999. Bayesian map learning in dynamic environments. Advances in Neural Information Processing Systems (NIPS). MIT Press.

[7] Kim C., Sakthivel R., and Chung, W. K. 2008. Unscented FastSLAM: a robust and efficient solution to the SLAM problem. IEEE Transactions of Robotics. Vol. 24(4). pp. 808–820.

[8] Montemerlo M. 2003. FastSLAM: A factored solution to the simultaneous localization and mapping problem

with unknown data association. Ph.D. dissertation. Carnegie Mellon Univ., Pittsburgh, PA.

[9] Montemerlo M., Thrun S., Roller D. and Wegbreit B. 2003. FastSLAM2.0: an improved particle filter algorithm for simultaneous localization and mapping that provably converges, Proc. of the 18th Int. Joint Conf. on Artificial Intelligence, San Francisco, CA, USA. pp. 1151–1156.

[10] Razali S., Watanabe K. and Meayama S. 2012. Unscented transformation for a FastSLAM framework. Int. Conf. on Artificial Life and Robotics (AROB 17th 2012), Oita, Japan. pp. 196–199.

[11] Van der Merwe R. 2004. Sigma point Kalman filter for probabilistic inference in dynamic state-space models. Ph.D. dissertation, Oregon Health & Science University, Portland.

[12] Julier S. J. 2003. The spherical simplex unscented transformation, Proc. of the American Control Conference, Denver, Colorado, USA. pp. 2430–2434.

[13] Julier S. J. and Uhlmann, J. K. 2002. The Scaled Unscented Transformation. Proc. of the American Control Conference, Anchorage, Alaska, USA. pp. 4555–4559.

[14] Bailey T. 2002. Mobile robot localization and mapping in extensive outdoor environments. Ph.D. dissertation, University of Sydney, Australian Center for Field Robotics.