Robotica: page 1 of 21. © The Author(s), 2020. Published by Cambridge University Press.
doi:10.1017/S026357472000065X

Robust Square-Root Cubature FastSLAM with Genetic Operators

Ramazan Havangi*

Faculty of Electrical and Computer Engineering, University of Birjand, Birjand, Iran

(Accepted June 19, 2020)

SUMMARY
An improved FastSLAM based on the robust square-root cubature Kalman filter (RSRCKF) with partial genetic resampling is proposed in this paper. In the proposed method, RSRCKF is used to design the proposal distribution of FastSLAM and to estimate environment landmarks. The proposed method does not require a priori knowledge of the noise statistics. In addition, to increase diversity, it uses the genetic operators-based strategy to further improve the particle diversity. In fact, a partial genetic resampling operation is carried out to maintain the diversity of particles. The proposed method is compared with other methods via simulation and experimental data. It can be seen from the results that the proposed method provides significantly more accurate and robust estimation results compared with other methods even with fewer particles and unknown a priori. In addition, the consistency of the proposed method is better than that of other methods.

KEYWORDS: SLAM; H-infinity filtering; Particle filter; Cubature Kalman filter; UFastSLAM.

1. Introduction

Simultaneous localization and mapping (SLAM) addresses the problem of incrementally building a feature map of the unknown environment while concurrently using this map to localize the robot itself.[1,2] In the recent years, FastSLAM method has attracted enormous attention from researchers as an efficient new solution to the SLAM problem in mobile robotics.[3]

In FastSLAM, the particle filter is used to estimate the robot pose (position and heading of robot) and extended Kalman filter (EKF) is used to estimate the location of landmarks.[4] The advantage of this algorithm is that the data association decisions can be determined on a per-particle basis, and hence different particles can be associated with different landmarks.[3] The other advantage of FastSLAM is that particle filters can cope with nonlinear and non-Gaussian robot motion models.[4]

However, FastSLAM suffers some drawbacks that are derivation of the Jacobian matrices and the linear approximations of the nonlinear functions. In addition, it cannot produce a consistent estimation in a long term. This is because each particle that carries the robot pose estimation information implicitly records a pose history in the statistics of its associated. In addition, during resampling process in FastSLAM, only a high weighted particle is selected and replicated while the low weighted particles are eliminated.[5] Therefore, each time a particle is lost, a whole map hypothesis is lost, and this causes the depletion in historical information. Hence, the overall map statistics will suffer degradation.[6]

For solving these problems, a number of authors have proposed UFastSLAM.[6,7] In UFastSLAM, the linearization process with Jacobian calculations is removed by applying the unscented transformation (UT).[8] In addition, UT is used in the prediction stage, and a better proposal distribution is provided by unscented Kalman filter (UKF) without the accumulation of Jacobian matrices and the linear approximations of nonlinear functions in the measurement update step. UKF is also used to update the mean and covariance of the feature and to initialize new features.[7] The main

* Corresponding author. E-mail: Havangi@birjand.ac.ir

advantage of UFastSLAM is that it does not use the derivation of the Jacobian matrices and the linear approximations to the nonlinear functions.

However, UFastSLAM only can be achieved consistently for longer time periods.[6] There are three main sources for inconsistency of UFastSLAM: first, the performance of UFastSLAM depends on correct a priori knowledge of the process and measurement noise that are unknown in real-life applications. An incorrect a priori knowledge may seriously degrade the performance of UKF[9–12] and consequently UFastSLAM. It can even lead to practical divergence and inconsistency. To overcome this problem, the author of this paper proposed an adaptive Neuro-Fuzzy method to dynamically adjust the noise statistics in previous work.[13]

Second, three scalar scaling parameters are employed by the UKF, and they are always selected based on the nonlinearity of the motion or observation models.[14] In addition, the choice of parameter value will affect estimated precision of the filter. However, there is no way to accurately estimate the nonlinearity level of nonlinear model, so it is impossible to find the optimal values for these parameters.[15]

Finally, the optimal choice of the resampling strategy is vital to improve the accuracy and consistency of UFastSLAM. The resampling step leads to sample impoverishment because the particles with high weights are statistically selected many times.[16, 17] In addition, whenever resampling step is performed, an entire pose history and map hypothesis is lost forever. This leads to a loss of diversity among the particles representing past poses and consequently erodes the statistics of the landmark estimates conditioned on these past poses.[18] As time goes, the number of distinct particles decreases and consequently the covariance of samples will decrease. This will cause inconsistency of filter.[19, 20]

To overcome the above-mentioned limitations, there are many attempts that apply some biological evolution algorithms. Genetic algorithm[21] and particle swarm optimization (PSO)[22–24] are two commonly used methods to maintain the diversity of particles before resampling step. In Ref. [24], PSO is adopted to solve the impoverishment problems and particle depletion. In this paper, the particles optimize after resampling and directly maintain the sufficient number of particles which contributes to estimating the robot pose accurately.

An improved FastSLAM algorithm-based robust square-root cubature Kalman filter (RSRCKF) with partial genetic resampling is proposed in this paper to enhance the performance of UFastSLAM. The main contributions are as follows:

1) The proposed algorithm uses third-degree cubature rule to design an optimal proposal distribution of the particle filter and to estimate the Gaussian densities of the feature landmarks. In fact, the proposed method uses the RSRCKF to generate the importance density and estimation of features. Also, this algorithm can work in unknown statistical noise behavior and thus it is more robust. In addition, beside the merit of reducing the computational complexity, the proposed algorithm has other advantages such as consistently increased numerical stability and better performance because all resulting covariance matrices are guaranteed to stay semi-positive definite.
2) In the proposed algorithm, we present partial genetic resampling. In this resampling scheme, the particles recombine by using genetic operators, and unlike the traditional resampling schemes, there is no duplication and elimination of particles. It is carried out only when the effective particles are less than the threshold.

The rest of this paper is organized as follows. Section 2 introduces the basic FastSLAM algorithm and the required background. Section 3 provides the framework of the proposed method. In Section 4, the simulation and experimental results are shown based on the simulator and benchmark dataset collected in Victoria Park.

2. Background

2.1. The basic FastSLAM

FastSLAM is an efficient algorithm for the SLAM problem, which decomposes the full SLAM posterior into a product of a robot path posterior and N landmark posteriors conditioned on the robot path:

$$p(s^t, \Theta | z^t, u^t, n^t) = p(s^t | z^t, u^t, n^t) \prod_{n=1}^{N} p(\theta_{n_t} | s^t, z^t, u^t, n^t) \quad (1)$$

where $s^t = \{s_1, ..., s_t\}$ is a robot path, and $z^t = \{z_1, ..., z_t\}$ and $u^t = \{u_1, ..., u_t\}$ are the measurements and controls up to time t, respectively. Here, data association $n^t = \{n_1, ..., n_t\}$ represents the mapping between map points and observation in z^t. Each particle in FastSLAM is as the following form:[4]

$$\left\{s^{t,[m]}, \mu_{1,t}^{[m]}, \Sigma_{1,t}^{[m]}, ..., \mu_{N,t}^{[m]}, \Sigma_{N,t}^{[m]}\right\} \tag{2}$$

where $[m]$ indicates the index of the particle, $s^{t,[m]}$ is the m-th particle's path estimate, and $\mu_{N,t}^{[m]}$, $\Sigma_{N,t}^{[m]}$ are the mean and the covariance of the Gaussian distribution representing the n-th feature location conditioned on the path $s^{t,[m]}$. In FastSLAM, a particle filter is used to compute the posterior over vehicle paths. For each feature in the map, this method uses a separate estimator EKF over its location.

2.2. *Cubature rule*

The most important step for Bayesian filtering in Gaussian domain is to calculate nonlinear transition density of the Gaussian prior. It can be written as:[25]

$$I = \int g(x)\mathrm{N}(x, P_x)dx \tag{3}$$

where $x \in R^{n_x}$, g is the nonlinear function, and $\mathrm{N}(x, P)$ indicates the n_x dimensional Gaussian prior density of the random variable of x. The CKF uses the third-degree spherical cubature rule to numerically calculate the integral I with $2n_x$ equal weighted cubature points as follows:[25–27]

$$I = \frac{1}{2n_x}\sum_{j=1}^{2n_x} g\left(\sqrt{P_x}\xi_j + x\right) \tag{4}$$

where $\sqrt{P_x}$ is a square-root factor of the covariance P_x, defined $\sqrt{P_x}\sqrt{P_x}^T = P_x$, and the cubature point ξ_j is given by:[25,26]

$$\xi_j = \begin{cases} \sqrt{n_x}e_i^T & i = 1, 2, \ldots, n_x \\ -\sqrt{n_x}e_i^T & i = n_x + 1, n_x + 2, \ldots, 2n_x \end{cases}$$

where $e_i^T \in R^{n_x}$ denotes the i-th column vector of the identity matrix $I_{n_x \times n_x}$.

2.3. *H_∞ extended filter*

To describe H_∞ extended filter (HEF), the state space model is considered as follows:

$$\begin{aligned} x_t &= f(x_{t-1}) + w_t \\ z_t &= h(x_t) + v_t \end{aligned} \tag{5}$$

The process noise w_t and the measurement noise v_t are assumed to be energy-bonded l_2 signals whose statistical properties are unknown:[28,29]

$$\|w_t\|_2^2 < \infty \; \|v_t\|_2^2 < \infty \tag{6}$$

For the system description in (5), the standard energy cost function J for the H_∞ filter optimization can be stated as:[28,29]

$$J = \frac{\sum_{t=0}^{t} \left\|z_t - \hat{z}_t\right\|_2^2}{\left\|x_0 - \hat{x}_0\right\|_{P_{0|0}^{-1}}^2 + \sum_{t=0}^{t} \|w_t\|_{Q_t^{-1}}^2 + \sum_{t=0}^{t} \|v_t\|_{R_t^{-1}}^2} < \gamma \tag{7}$$

where γ represents the user-defined values of the robustness bound, $P_{0|0}$, Q_t, and R_t are the weighting matrices for the initial condition, the process noise, and the measurement noise, respectively. The objective of H_∞ estimation is to minimize the maximum value of J as follows:[28]

$$\min_{\hat{\mathrm{x}}_\mathrm{t}} \max_{\mathrm{v_t, w_t, x_{0|0}}} J < \gamma \tag{8}$$

and the corresponding sequential HEF formulation can be stated as follows:

$$\hat{x}_{t|t-1} = f(x_{t-1|t-1})$$

$$P_{t|t-1} = F_t P_{t-1|t-1} F_t^T + Q_t, \quad F_t = \frac{\partial f}{\partial x} \tag{9}$$

$$K_t = P_t^- H_t^T (H_t P_{t|t-1} H_t^T + R_t)^{-1}, \quad H_t = \frac{\partial \mathrm{h}}{\partial \mathrm{x}}$$

$$R_{e,t} = \begin{bmatrix} R_t & 0 \\ 0 & -\gamma I \end{bmatrix} + \begin{bmatrix} H_t \\ I \end{bmatrix} P_t^- \begin{bmatrix} H_t^T & I \end{bmatrix}$$

$$\hat{x}_{t|t} = \hat{x}_{t|t-1} + K_t \left(z_t - h\left(\hat{x}_{t|t-1}^- \right) \right) \tag{10}$$

$$P_{t|t} = P_{t|t-1} - P_{t|t-1} \begin{bmatrix} H_t^T & I \end{bmatrix} R_{e,t}^{-1} \begin{bmatrix} H_t \\ I \end{bmatrix} P_{t|t-1}$$

where K_k is the gain filter, and Q_t and R_t are the covariance of the process noise and the measurement noise, respectively.

3. Proposed Method

In the proposed method, the nonlinear robot motion and nonlinear environment measurement model are defined as follows:

$$\begin{aligned} s_t &= f(s_{t-1}, u_{t-1} + w_t) \\ z_t &= h\left(s_t, \mu_{t-1}^{[m]}\right) + v_t \end{aligned} \tag{11}$$

where s_t is the robot pose, z_t is the observation vector, f and h are the nonlinear robot dynamics and the observation model, respectively, u_t is the control input, w_t is control noise with covariance Q_t, v_t is observation noise with covariance R_t, $\mu_t^{[m]}$ is the m-th revisited landmark state. The FastSLAM methods require Jacobians and assume that the statistical noise properties are known during the state estimation. In the proposed method, neither the Jacobians nor the statistical noise properties are required for state estimation. The proposed method fuses H_∞ filter and SRCKF to obtain a filter that will have the desirable properties of both filters. In addition, to increase diversity, it uses the genetic resampling. This algorithm consists of sampling strategy, update of revisited landmark, calculation of the importance weight, and partial genetic resampling.

3.1. Sampling strategy

The crucial step in designing a particle filter-based SLAM (FastSLAM) is the choice of proposal distribution. The simplest choice is often to sample from the transition model. However, in many cases, such proposals result in poor filter performance due to a mismatch in the areas of high probability between the transition and observation distributions. The optimal importance distribution choice of importance distribution for each particle is the conditional posterior given both the previous state and the new observation.[30] Therefore, the optimal distribution for FastSLAM is as follows:

$$q\left(s_t^{[m]} | s^{t-1,[m]}, z^t, u^t, n^t\right)$$

It is designed with intelligent robust square-root cubature particle filter (RSRCPF). The intelligent RSRCPF is combined of SRCKF and particle filter with partial genetic resampling and is used to compute the mean and covariance of the vehicle state.

For sampling from proposal distribution, the first step is to form an augmented state and augmented covariance matrix by appending the mean and covariance of process noise vector. Assuming the mean of process noise is zero, the mean and covariance of augmented state are as follows:

$$s_{t-1|t-1}^{a[m]} = \begin{bmatrix} s_{t-1|t-1}^{[m]} \\ 0 \end{bmatrix} \quad P_{t-1|t-1}^{a[m]} = \begin{bmatrix} P_{t-1|t-1}^{[m]} & 0 \\ 0 & Q_t \end{bmatrix} \tag{12}$$

where $s_{t-1|t-1}^{[m]}$ and $P_{t-1|t-1}^{[m]}$ are the previous mean and covariance of the robot pose, respectively, Q_t is the process noise covariance, $s_{t-1|t-1}^{a[m]}$ is the mean of the augmented state, and $P_{t-1|t-1}^{a[m]}$ is the covariance of augmented state. Use the Cholesky factor of $P_{t-1|t-1}^{a[m]}$ as follows:

$$S_{t-1|t-1}^{a[m]} = chol\left(P_{t-1|t-1}^{a[m]}\right) \tag{13}$$

Then, $2n_a$ symmetry cubature points is calculated by

$$\delta_{t-1|t-1}^{a[i][m]} = s_{t-1|t-1}^{a[m]} + S_{t-1|t-1}^{a[m]}\zeta_{\mathrm{i}} \qquad i = 1, \ldots 2n_a \tag{14}$$

where ζ_{i} is the i-th element of $2n_a$ cubature points as follows:

$$\left\{\begin{bmatrix}1\\0\\\vdots\\0\end{bmatrix}, \begin{bmatrix}0\\1\\\vdots\\0\end{bmatrix}, \ldots, \begin{bmatrix}0\\0\\\vdots\\1\end{bmatrix}, \begin{bmatrix}-1\\0\\\vdots\\0\end{bmatrix}, \begin{bmatrix}0\\-1\\\vdots\\0\end{bmatrix}, \ldots, \begin{bmatrix}0\\0\\\vdots\\-1\end{bmatrix}\right\}$$

where n_a is the size of the state vector $s_{t-1|t-1}^{a}$. Each cubature point can be decomposed to the robot state and the control components:

$$\delta_{t-1|t-1}^{a[i][m]} = \begin{bmatrix}\delta_{t-1|t-1}^{[i][m]} & \delta_t^{u[i][m]}\end{bmatrix}^T \tag{15}$$

These cubature points pass through the robot motion model f as:

$$\hat{s}_t^{[i][m]} = f\left(\delta_{t-1|t-1}^{[i][m]}, u_t + \delta_t^{u[i][m]}\right) \tag{16}$$

The predicted mean $s_{t|t-1}$ is calculated from the transformed cubature point set $\hat{s}_t^{[i][m]}$ based on cubature as:

$$s_{t|t-1}^{[m]} = \frac{1}{2n_a}\sum_{i=1}^{2n_a}\hat{s}_t^{[i][m]} \tag{17}$$

To predict the square-root factor of the covariance of the robot state $S_{t|t-1}^{[m]}$, an error matrix E_r is defined as follows:

$$E_r = \hat{s}_t^{[1:2n][m]} - s_{t|t-1}^{[m]}$$

The square root of the covariance matrix $S_{t|t-1}^{[m]}$ is predicted as:

$$S_{t|t-1}^{[m]} = qr\left\{\left[\frac{1}{\sqrt{2n_a}}E_r\right]^T, 0\right\} \tag{18}$$

When a feature landmark is revisited by the robot, the particle state s_k and its covariance squared root factor S_k can be updated. To update them, s_k and S_k are augmented with the revisited landmark first:

$$b = \begin{bmatrix} s_{k|k-1}^{[i]} \\ \mu_{k-1}^{[i][m]} \end{bmatrix}, \; P_b = \begin{bmatrix} P_{k|k-1}^{[i]} & 0 \\ 0 & \Sigma_{k-1}^{[i][m]} \end{bmatrix} \tag{19}$$

where b is the augmented prediction state and p_b is the augmented prediction covariance, respectively. Then the cubature point set for the augmented density is calculated as:

$$\delta_{t-1|t-1}^{*a[i][m]} = b + chol\left(P_b\right)\zeta_{\mathrm{i}} \qquad i = 1, \ldots 2n_a \tag{20}$$

Each cubature point $\delta^{*a[i][m]}_{t-1|t-1}$can be expressed by the compound of the robot state and the landmark state components as follows:

$$\delta^{*}_{t-1|t-1} = \left[\bar{s}_t, \mu_{t,t-1}\right]^T$$

and the transformed cubature point set is calculated based on the nonlinear measurement function h, given by:

$$\zeta_t^{[i][m]} = h\left(\bar{s}_t^{[i][m]}, \mu_{t,t-1}^{[m]}\right)$$

Consequently, the predicted measurement for the revisited landmark is calculated with the transformed cubature point set as:

$$\bar{z}_t^{[m]} = \frac{1}{2n_b}\sum_{i=0}^{2n_b} \zeta_t^{[i][m]}$$

The measurement innovation covariance $S_{zz}^{[m]}$ and the cross covariance $P_{\delta\upsilon}^{[m]}$ between the robot state s and the measurement z are:

$$S_{zz}^{[m]} = \frac{1}{2n_b}\sum_{i=0}^{2n_b}\left(\zeta_t^{[i][m]} - \bar{z}_t^{[m]}\right)\left(\zeta_t^{[i][m]} - \bar{z}_t^{[m]}\right)^T + R$$

$$P_{\delta\upsilon}^{[m]} = \frac{1}{2n_b}\sum_{i=0}^{2n_b}\left(\bar{s}_t^{[i][m]} - \delta_{t-1}^{[i][m]}\right)\left(\zeta_t^{[i][m]} - \bar{z}_t^{[m]}\right)^T \tag{21}$$

The mean and square root of covariance of the vehicle are updated as:

$$s_{t|t}^{[m]} = s_{t|t-1}^{[m]} + K_t^{[m]}\left(z_t - \bar{z}_t^{[m]}\right) \tag{22}$$

$$P_{t|t}^{[m]} = S_{t|t-1}^{[m]}\left(S_{t|t-1}^{[m]}\right)^T - \left[P_{\delta\upsilon}^{[m]} \quad S_{t|t-1}^{[m]}\left(S_{t|t-1}^{[m]}\right)^T\right] R_{e,k}^{-1}\left[P_{\delta\upsilon}^{[m]} \quad S_{t|t-1}^{[m]}\left(S_{t|t-1}^{[m]}\right)^T\right]^T \tag{23}$$

$$S_{t|t}^{[m]} = chol\left(P_{t|t}^{[m]}\right)$$

where $K_t^{[m]}$ and $R_{e,t}$ are:

$$K_t^{[m]} = P_{\delta\upsilon}^{[m]}\left(S_{zz}^{[m]}\right)^{-1} \tag{24}$$

$$R_{e,t} = \begin{bmatrix} R_t + P_{\delta\upsilon}^{[m]} & [P_{\delta\upsilon}^{[m]}]^T \\ P_{\delta\upsilon}^{[m]} & -\zeta I + P_{t|-1} \end{bmatrix} \tag{25}$$

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

$$s_t^{[m]} \sim N\left(s_t; s_{t|t}^{[m]}, P_{t|t}^{[m]}\right) \tag{26}$$

where the covariance matrix of the robot pose is as follows:

$$P_{t|t}^{[m]} = \left(S_{t|t}^{[m]}\right)^T S_{t|t}^{[m]}. \tag{27}$$

The importance weight of particles w_t is determined as follows:

$$w_t^{[m]} = w_{t-1}^{[m]}\frac{p\left(z_t|s_t^{[m]}\right)p\left(s_t^{[m]}|s_{t-1}^{[m]}, u_t\right)}{N\left(s_{t|t}; s_{t|t}^{[m]}, P_{t|t}^{[m]}\right)}. \tag{28}$$

3.2. Update of revisited landmark

If a landmark is revisited at time t, a set of sigma points are sampled from the revisited feature as:

$$\chi^{[i][m]} = \mu^{[m]}_{n_t,t-1} + \Omega^{[m]}_{n_t,t-1}\zeta_i, \qquad (i = 1, \ldots, 2n_l) \tag{29}$$

where n_l is the dimension of landmarks in the map, $\mu^{[m]}_{n_t,t-1}$ and $\Sigma^{[m]}_{n_t,t-1} = (\Omega^{[m]}_{n_t,t-1})^T\Omega^{[m]}_{n_t,t-1}$ are the mean and covariance matrix $n\text{-}th$ feature that is registered in feature initialization step. By passing the cubature points $\chi^{[i][m]}$ through the nonlinear measurement function h, the predicted measurement $\hat{z}^{[m]}_t$ is calculated as follows:

$$\bar{Z}^{[i][m]}_t = h\left(\chi^{[i][m]}, s^{[m]}_t\right), (i = 1, \ldots, 2n_l)$$

$$\hat{z}^{[m]}_t = \frac{1}{2n_l}\sum_{i=0}^{2n_l}\bar{Z}^{[i][m]}_t \tag{30}$$

As a result, the covariance matrixes for the $m\text{-}th$ revisited landmark are calculated with cubature points are as:

$$S^{[m]}_{zz} = \frac{1}{2n_l}\sum_{i=0}^{2n_l}\left(\bar{Z}^{[i][m]}_t - \hat{z}^{[m]}_t\right)\left(\bar{Z}^{[i][m]}_t - \hat{z}^{[m]}_t\right)^T + R$$

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

The mean $\mu^{[m]}_{n_t,t}$ and the square root of covariance $\Omega^{[m]}_{n,t}$ of the $n\text{-}th$ feature are updated by:

$$\mu^{[m]}_{n_t,t} = \mu^{[m]}_{n_t,t-1} + \bar{K}^{[m]}_t\left(z_t - \hat{z}^{[m]}_t\right) \tag{31}$$

$$\begin{aligned}\Sigma^{[m]}_{n_t,t} &= \Sigma^{[m]}_{n_t,t-1} - \Sigma^{[m]}_{n_t,t-1}\begin{bmatrix} G^T_{\theta_{n_t}} & I\end{bmatrix} R^{-1}_{e,t}\begin{bmatrix} G_{\theta_{n_t}} \\ I\end{bmatrix}\Sigma^{[m]}_{n_t,t-1} \\ \Omega^{[m]}_{n,t} &= chol\left(\Sigma^{[m]}_{n_t,t}\right)\end{aligned} \tag{32}$$

where the gain $\bar{K}^{[m]}_t$ and $R_{e,t}$ are determined as:

$$\bar{K}^{[m]}_t = \bar{\Sigma}^{[m]}_{t\chi Z}\left(S^{[m]}_{zz}\right)^{-1} \tag{33}$$

$$R_{e,t} = \begin{bmatrix} R & 0 \\ 0 & -\zeta I\end{bmatrix} + \begin{bmatrix} G_{\theta_{n_t}} \\ I\end{bmatrix}\Sigma^{[m]}_{n_t,t-1}\begin{bmatrix} G^T_{\theta_{n_t}} & I\end{bmatrix} \tag{34}$$

3.3. Partial genetic resampling

In the proposed method, a combination of RSRCKF and particle filter is used to estimate the vehicle path. However, the phenomenon of particle degradation is inevitable. The resampling process is performed to reduce the effects of the degeneracy. During the resampling step, the small-weight particles might end up with no children and the large-weight particles have a large number of children.[16,17] Although the resampling step reduces the effects of the degeneracy problem, it decreases the diversity of particles.[17]

In order to maintain the particle diversity in the proposed method, the crossover and mutation are applied on particles before the resampling step. For this purpose, the particles are divided into the small-weight particles C_L and large-weight particles C_H. In order to separate the small-weight particles from other large-weight ones, the weight of each particle is compared with a threshold ω_T.

After separating the particles, the crossover operator is performed on small-weight particles. In fact, the small-weight particles are modified using the information of the large-weight particles through the crossover operator. There are many crossover operators. In this paper, arithmetic crossover is selected to modify the small-weight particles. Assume that $x_t^{(l)}$ and $x_t^{(h)}$ respectively represent the particles from C_L and C_H. The modified small-weight particle is denoted by $x_t^{\prime[l]}$ as follows:

$$x_t^{\prime(l)} = \alpha x_t^{(l)} + (1-\alpha) x_t^{(h)} \tag{35}$$

in which $l = 1 \ldots N_L$ and $h = 1 \ldots N_H$. N_L and N_H respectively represent the number of particles contained in C_L and C_H. For each $x_t^{\prime(l)}$, $x_t^{(h)}$ is randomly selected from C_H. $\alpha \in [0, 1]$ is a parameter which determines how much information for $x_t^{[l]}$ transferred to $x_t^{[h]}$. The greater α is the more information will be transferred from $x_t^{[l]}$ to $x_t^{[h]}$. To increase the diversity of particles, a mutation strategy is applied on the particle $x_t^{\prime[h]}$ according to mutation probability p_m:

$$x_t^{[l]} = \begin{cases} x_t^{\prime[l]} + \beta & \text{if } r_1 \le p_m \\ x_t^{\prime[l]} & \text{if } r_1 > p_m \end{cases} \tag{36}$$

where vector β is taken from a uniform distribution, r_1 is the random variable for $x_t^{\prime[l]}$ that is drawn from the uniform distribution on [0, 1] and p_m denotes the mutation probability. The crossover and mutation operators can move particles to the region of high likelihood. The choice of p_m and α has direct impact on the moving speed. The weighting of the modified particles $x_t^{\prime(l)}$ are calculated as:

$$w_t^{(l)} = w_{t-1}^{(l)} \frac{p\left(y_t | x_t^{\prime(l)}\right) p\left(x_t^{\prime(l)} | x_{t-1}^{\prime(l)}\right)}{N\left(x_t^l;\, x_{t|t}^{\prime(l)},\, P_{t|t}^{l(i)}\right)} \tag{37}$$

After crossover and mutation, whenever effective number of particles N_{eff} is below a predefined threshold, the resampling procedure is performed. The effective number of particles N_{eff} is:

$$N_{eff} = \frac{1}{\sum_{m=1}^{N} \left(\omega_{\mathrm{t}}^{(m)}\right)^2} \tag{38}$$

where $\omega_{\mathrm{t}}^{(m)}$ is the normalized weight:

$$\omega_{\mathrm{t}}^{(m)} = \frac{\omega_{\mathrm{t}}^{(m)}}{\sum_{i=1}^{N} \omega_{\mathrm{t}}^{(i)}}$$

4. Simulation and Experimental Results

In this section, both numerical simulations and real-world dataset experiments are conducted to evaluate the performance of the proposed algorithm.

4.1. Simulation

The simulated environment is two-dimensional map that contains landmarks and robot path. Figure 1 shows the simulation environment map, where the star points (*) is the location of the landmarks. The robot starts from the origin of the global frame and moves at a speed 3 m/s and with a maximum steering angle of 30°. It detects nearby features with a laser sensor until it closes a big loop. The robot has 4 m wheel base and is equipped with a range-bearing sensor with a maximum range of 30 m and 180° frontal field-of-view. The control noise is ($\sigma_v = 0.2\,\mathrm{m/s}$, $\sigma_\gamma = 2^{\mathrm{o}}$).

The distances from landmarks to the robot and the angles between landmarks and the forward direction of the robot are obtained by a laser range finder. The measurement noise is ($\sigma_r = 0.1\mathrm{m}, \sigma_\theta = 1^{\mathrm{o}}$). Control frequency is 40 Hz and observation scans are obtained at 5 Hz.

A series of Monte Carlo comparison studies is conducted under different conditions, in order to evaluate the proposed method in comparison with other well-known algorithms including FastSLAM2.0 and UFastSLAM. To evaluate the performance of algorithms, the root mean squared

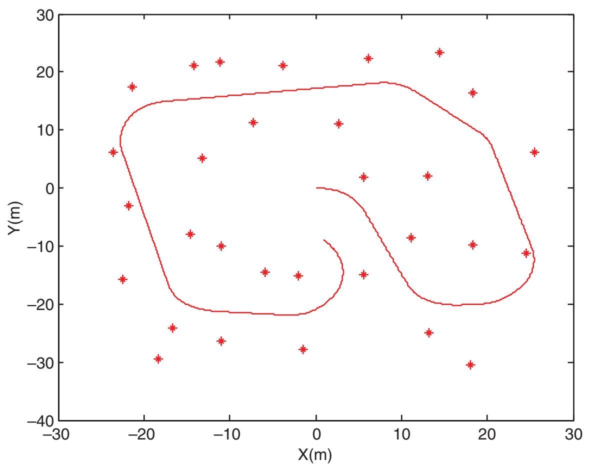

Fig. 1. True robot path and true landmarks.

error (RMSE), and the average normalized estimation error squared (NEES) is used. The RMSE provides a measure of accuracy and the NEES is a standard criterion for evaluating estimator consistency. For each algorithm, 20 independent Monte Carlo runs are conducted.

4.1.1. Performance under the known statistics of the noises. In this subsection, the performance of the proposed algorithm is evaluated while the statistical properties of noises are known a priori, that is, the measurement noise is $\sigma_r = 0.1$, $\sigma_\theta = 1$ and the control noise is $\sigma_v = 0.2$, $\sigma_\gamma = 2$. In this experiment, the number of particles is 50 and the results are obtained over 20 Monte Carlo runs. Figure 2 shows the estimation result of a typical simulation run of the algorithms. The result indicates that both the robot trajectory and features are estimated accurately by the proposed algorithm. The RMSE is adopted to evaluate the estimation accuracy of the FastSLAM algorithms quantitatively. The RMSE with respect to each time step in Fig. 3 and its mean and standard deviation are shown in Fig. 4.

It is observed that the proposed method has better accuracy than UFastSLAM and FAstSLAM2.0. This is because the genetic resampling causes increase of diversity and consequently increase accuracy in the proposed algorithm. In addition, RSRCKF is able to estimate the mean and covariance to a higher order of accuracy and consistency than the UKF in UFastSLAM and EKF in FastSLAM2.0.

To measure whether a filter is consistent, its estimate is used to compare with the probability density function (PDF) obtained from an ideal filter.[31] To verify the consistency, the NEES is used:

$$\varepsilon_t = \left(s_t - \hat{s}_{t|t}\right)^T \hat{P}_{t|t}^{-1} \left(s_t - \hat{s}_{t|t}\right)$$

where s_t *is* the ground truth, and $\hat{s}_{t|t}$ and $\hat{P}_{t|t}$ are the estimated mean and covariance, respectively. The consistency is evaluated by performing multiple Monte Carlo runs and computing the average NEES (ANEES). Given Nruns, ANEES is computed as:

$$\bar{\varepsilon}_t = \frac{1}{N} \sum_{i=1}^{N} \varepsilon_{it}$$

For the two-dimensional vehicle position with 20 Monte Carlo simulations, the two-sided 95% probability concentration region for $\bar{\varepsilon}_t$ is bounded by interval [1.3, 2.79] and algorithm is consistent if $\bar{\varepsilon}_t$ with probability 95% belong to [1.3, 2.79]. Figure 5 shows that the proposed method is consistent while UFastSLAM is inconsistent after 30 s. This is why diversity particles in the proposed method are more than that of UFastSLAM and FastSLAM2.0. Figure 6 shows the number of distinct particles, which are counted after every resampling. The result shows that the number of distinct particles in the proposed method is more than that of other methods. Hence, the consistency of the proposed method is more than that of other methods.

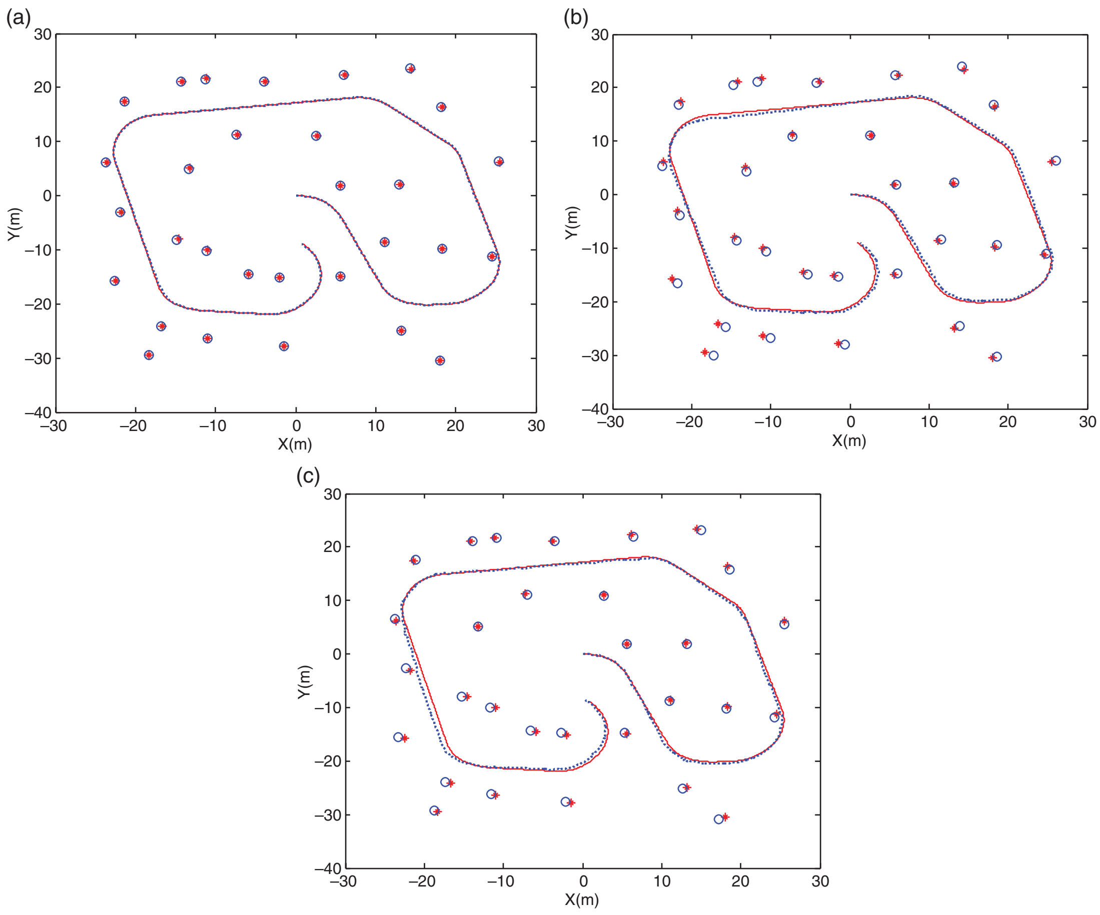

Fig. 2. Estimated robot path and estimated landmarks: The "…" is the estimated path, the "o" are the estimated landmark positions. (a) Proposed method; (b) UFastSLAM; (c) FastSLAM2.0.

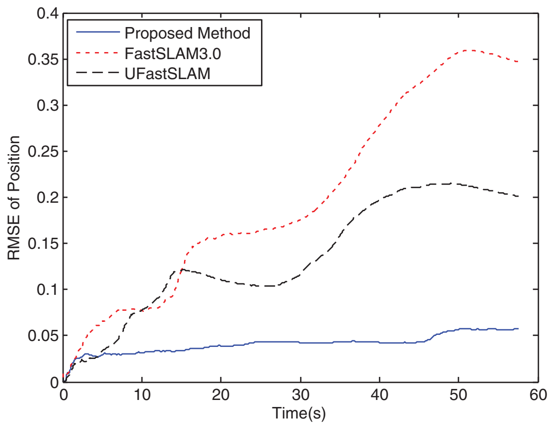

Fig. 3. RMSE of position with respect time.

4.1.2. Performance comparison with different particle numbers. The proposed algorithm is compared with others algorithms with the variety of the particles quantity. In the simulations, the number of particles is set to be 10, 30, 50, respectively. The RMSE is obtained using 20 Monte Carlo runs. The control noise level is $(\sigma_\nu = 0.2, \sigma_\gamma = 2)$, and the measurement noise level is $(\sigma_r = 0.1, \sigma_\theta = 1)$. With the given particle number level, the mean and the standard deviation of RMSE for the vehicle path estimation are calculated over 50 Monte Carlo runs for each SLAM algorithm. Figure 7 shows that the estimations of vehicle pose become more accurate for the three algorithms when more

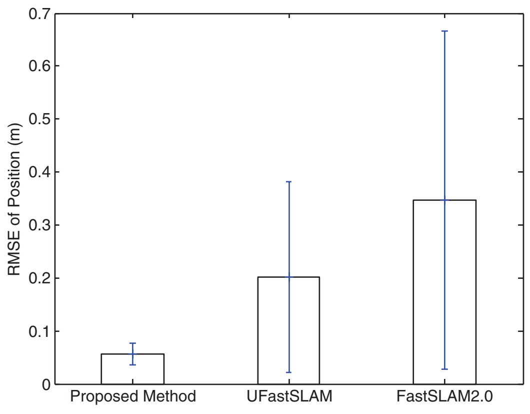

Fig. 4. Mean and standard deviation of RMSE.

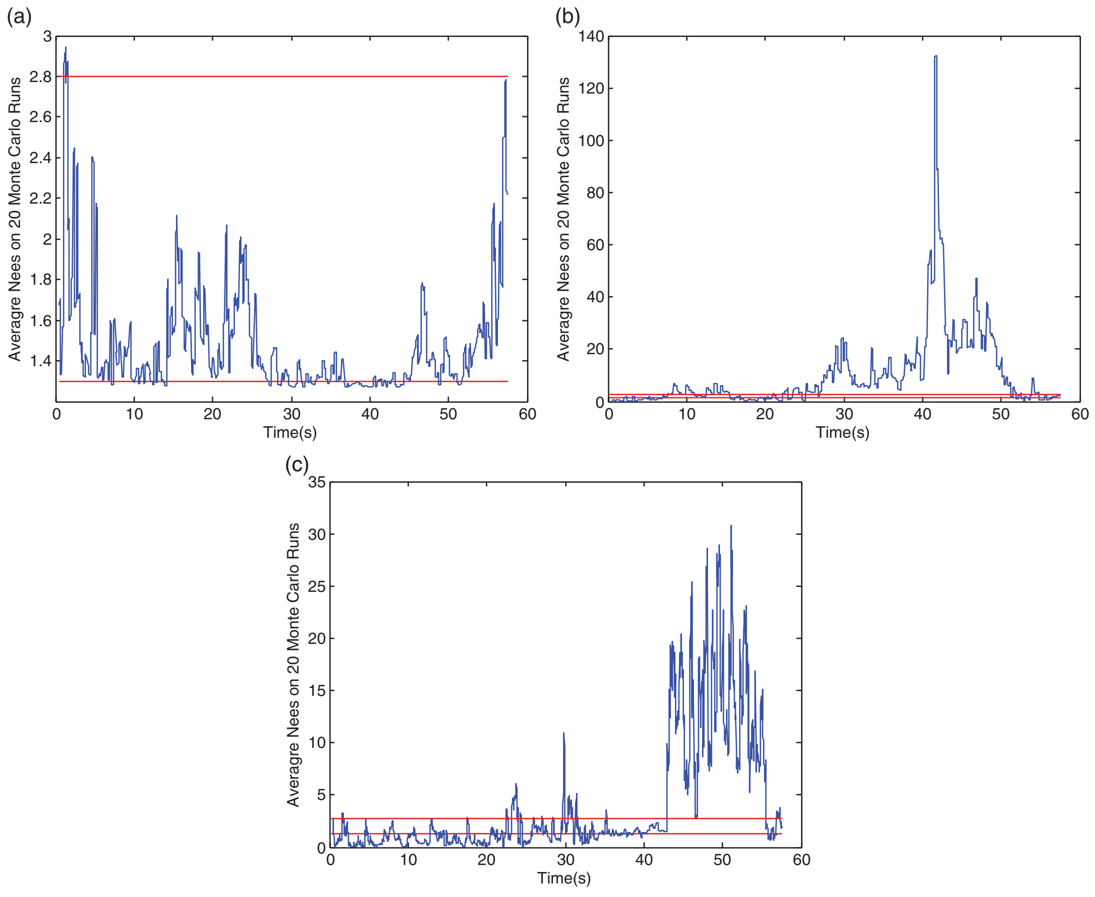

Fig. 5. Consistency: (a) Proposed method; (b) UFastSLAM; (c) FastSLAM2.0.

particles are used for SLAM algorithm. However, for same particles, the accuracy of proposed algorithm is better than that of other two algorithms. This is because the proposal distribution is designed by RSRCKF and the sampling accuracy of particles is improved. In addition, the genetic operators are used to reduce the degree of particles degeneracy and maintain the diversity of particles. The resampled particles gradually approximate the samples from the posterior PDF of the true state. Hence, in the proposed method, the fewer particles are used to accurately accomplish the SLAM and computational cost can be decreased.

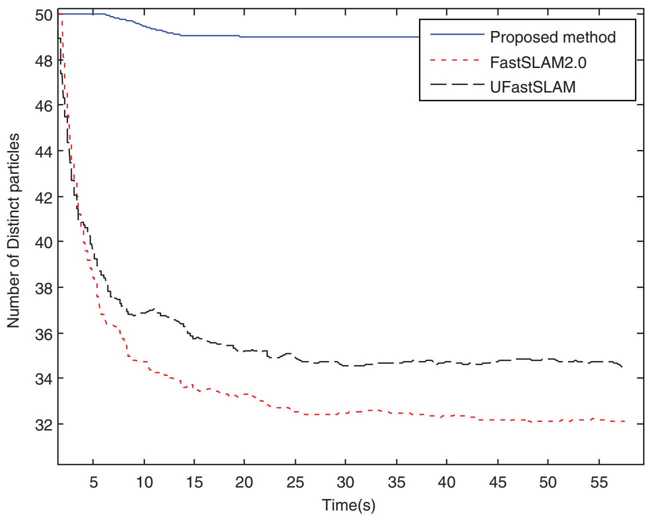

Fig. 6. Number of distinct particles.

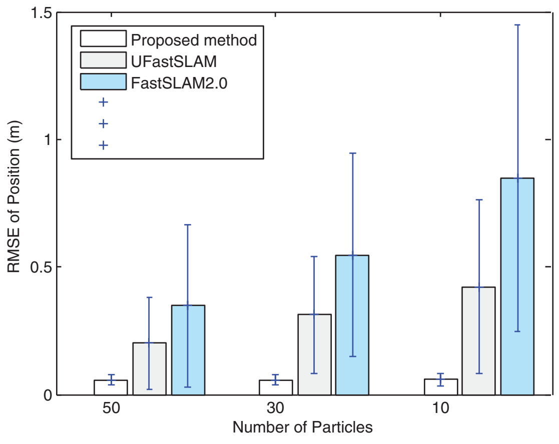

Fig. 7. Performance of algorithms with different numbers of particles.

4.1.3. Performance under the unknown statistics of the noises. To evaluate the performance proposed algorithm under the unknown statistics of the noises, the performance of it is compared while measurement noise is considered wrongly as ($\sigma_r = 0.05$, $\sigma_\theta = 0.5$) and the control noise is ($\sigma_v = 0.2\,\text{m/s}, \sigma_\gamma = 2^{\text{o}}$). The chosen particle number for algorithm is 50 and the results are shown in Fig. 8. It shows that when the robot position estimation is accurate, the position estimation of the landmarks is also accurate and vice versa. This is because these parameters depend on each other. The result indicates that the estimated vehicle path and estimated landmark coincide as closely as possible with the actual path and the actual positions features. In UFastSLAM and FastSLAM2.0, the error estimated vehicle path and estimated landmark are increased respectively.

The RMSE of position over time is presented in Fig. 9, and the mean and variance of it are shown in Fig. 10. Figure 11 shows the comparisons of the NEES for algorithms in this experiment. It can be seen that the proposed algorithm is consistent, but UFastSLAM becomes inconsistent after 5 s.

This is because that the proposed method achieves better numerical stability and smaller linearization error by using the square-root extension of the RSRCKF. Moreover, sufficient number of particles is always generated in FastSLAM and thus the accuracy of the estimation is increased. Finally, the proposed method does not require a priori knowledge of the noise statistics and deepened on only an assumption that the noises are bounded in certain energy level.

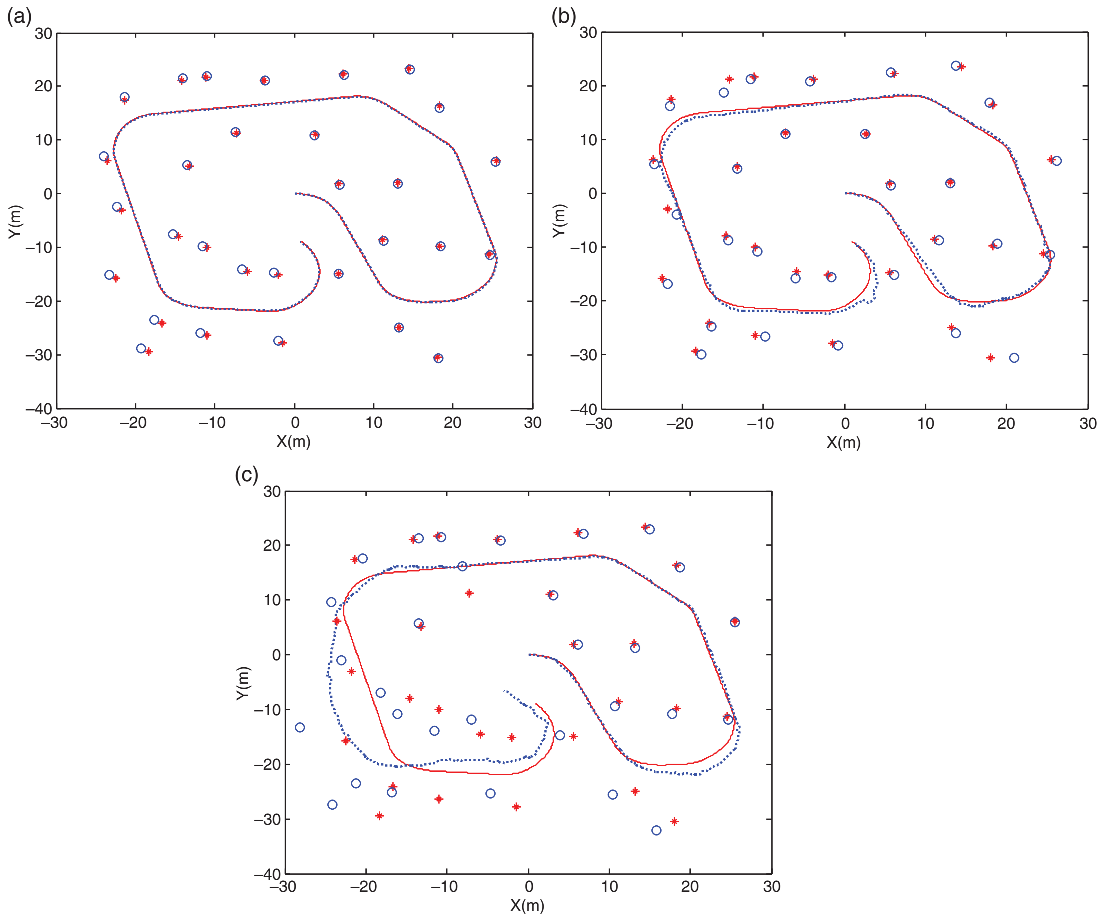

Fig. 8. Estimated robot path and estimated landmarks: The "…" is the estimated path, the "o" are the estimated landmark positions. (a) Proposed method; (b) UFastSLAM; (c) FastSLAM2.0.

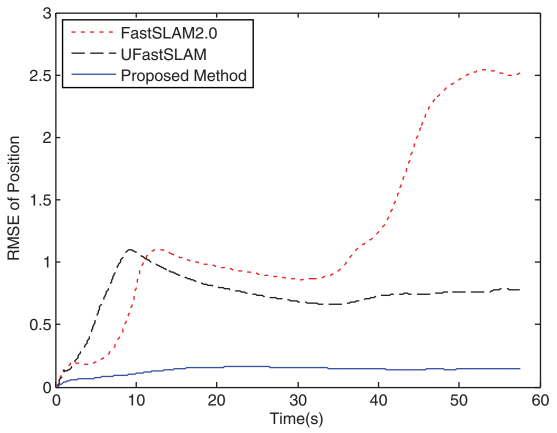

Fig. 9. RMSE of position with respect time.

4.1.4. Computational cost. The computational cost of the proposed method is analyzed using Matlab simulations on Intel(R) Core(TM) i5-4590 CPU@3.3 GHz PC. Table I demonstrates the composition of three algorithms. As shown, the CPU running time of three algorithms increases as the number of particles increases. Under specified particle number, the proposed method has better computational efficiency compared to other methods. The main advantage of the proposed method over the conventional methods from computational cost viewpoint is that to obtain the same estimation

Table I. Computational cost of algorithms.

	Processing time (s)		RMSE (m)	
Number of particles	**Proposed method**	**UFastSLAM**	**Proposed method**	**UFastSLAM**
50	67	56	0.056	0.2
30	47	36	0.057	0.31
10	21.4	13.2	0.059	0.43

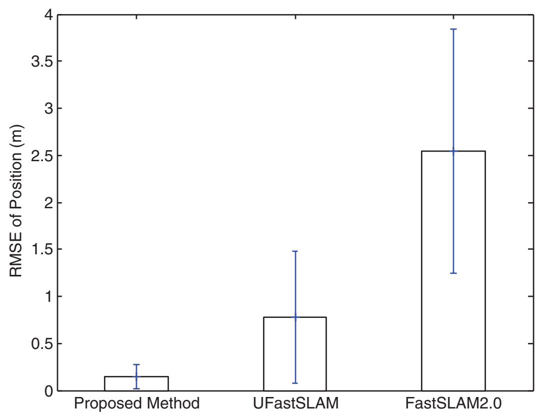

Fig. 10. Root mean square error.

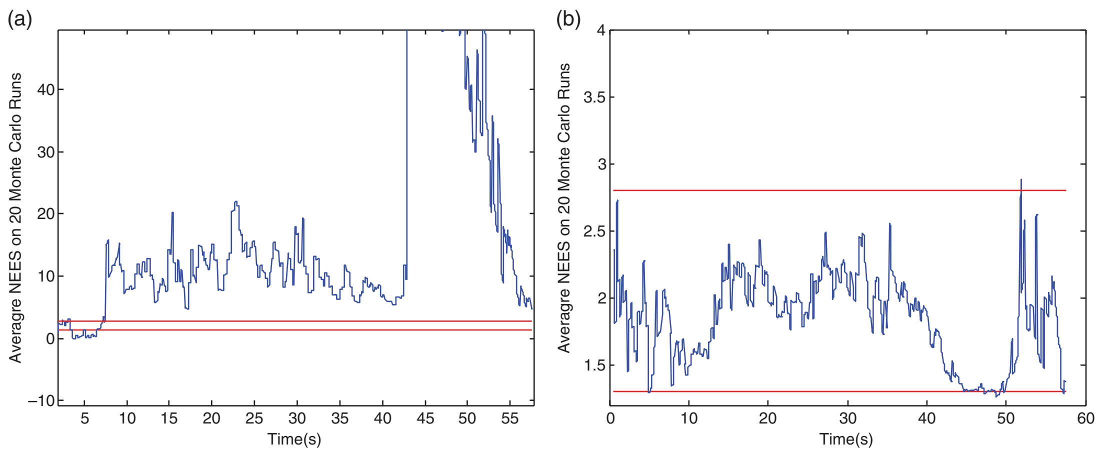

Fig. 11. Consistency: (a) UFastSLAM; (b) proposed method.

accuracy as other methods, lower number of particles is needed. Table I shows that when one-fifth of the particles of UFastSLAM (10 particles) is used, the estimated error is smaller than UFastSLAM. Therefore, the proposed method gains more accuracy with lower computational cost.

Moreover, the proposed resampling can also reduce the computational time, since the proposed resampling is only carried out when the effective particles is less than the threshold. Compared to standard resampling, the partial genetic resampling also presents better accuracy and computational efficiency. This is because it only conducts resampling on partial particles and merges the large-weight and small-weight particles to avoid repeatedly copying large-weight particles.

4.1.5. Performance of partial genetic resampling. The performance of the proposed algorithm is evaluated while the standard resampling is used instead of partial genetic resampling in it. This experiment shows that the performance improvement comes only from RSRCKF and shows the importance of the genetic resampling. Figure 12 shows the RMSE of estimations over time. Similar

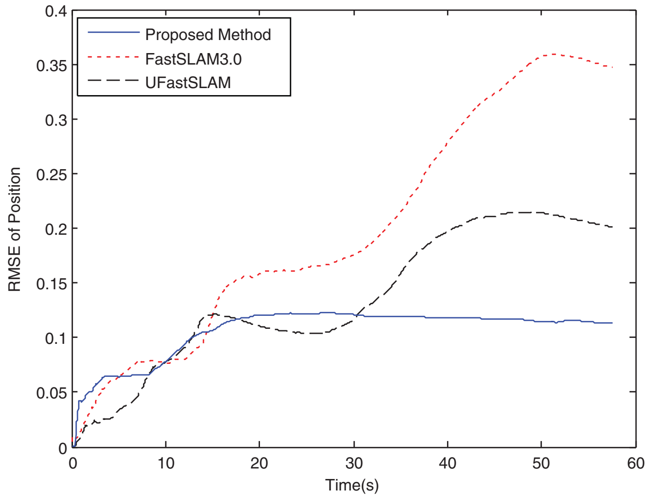

Fig. 12. RMSE of position.

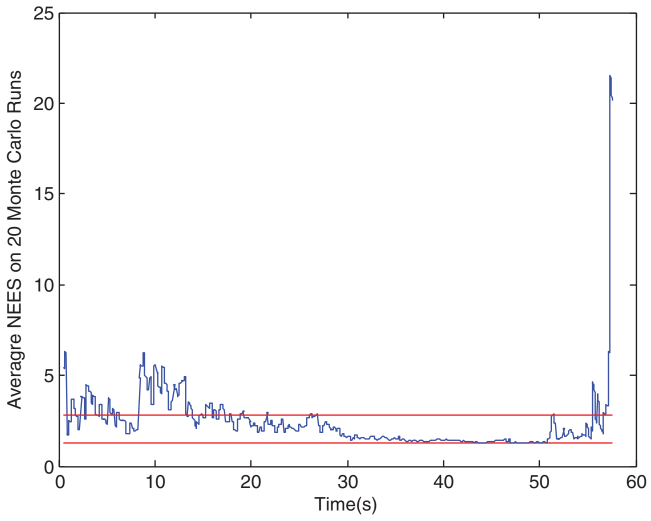

Fig. 13. Consistency of proposed method with standard resampling.

to partial previous cases, it is observed that the performance of the proposed method is better than that of other algorithms. However, the proposed algorithm becomes inconsistent after 55 s as shown in Fig. 13. This is why the proposed method employs the standard resampling similar to other methods. Comparing Figs. 13 and 3 shows that using the partial genetic resampling can help improve the estimation accuracy and consistency. In this way, the proposed resampling can better approximate the true state of vehicle.

4.1.6. Performance under non-Gaussian noise condition. In this subsection, performance of the proposed method is evaluated under non-Gaussian noise condition. For this purpose, the process noise is assumed to be Gaussian noise while the measurement noise is drawn from a Gamma distribution in this case. The results are shown in Fig. 14. It can be seen that the proposed method outperforms UFastSLAM. This is expected because of the fact that the proposed method outperforms FastSLAM2.0, UFastSLAM, and UKF-SLAM when the statistics of the noise processes are not exactly Gaussian and unknown by the algorithms. In UFastSLAM, non-Gaussian noise will have a significant effect because it is very sensitive to the nature of the noise. On the other hand, the H_∞ filter does not assume Gaussian noise and, thus, the proposed method is robust to variations in the nature of the noise. This robust property tends to decrease errors in the proposed method.

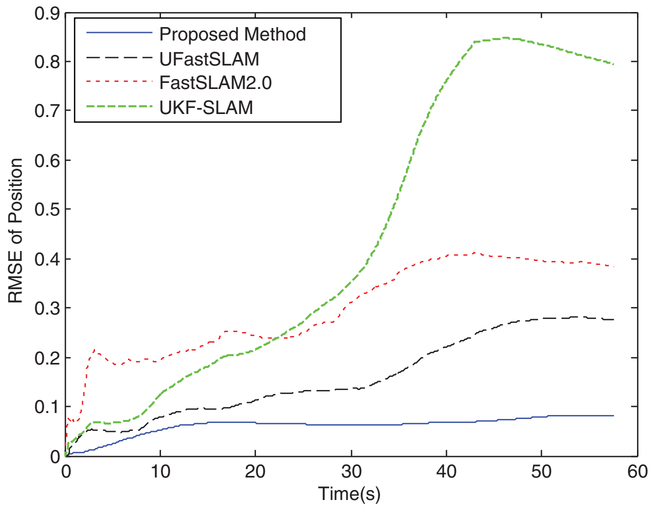

Fig. 14. RMSE of pose with respect to time.

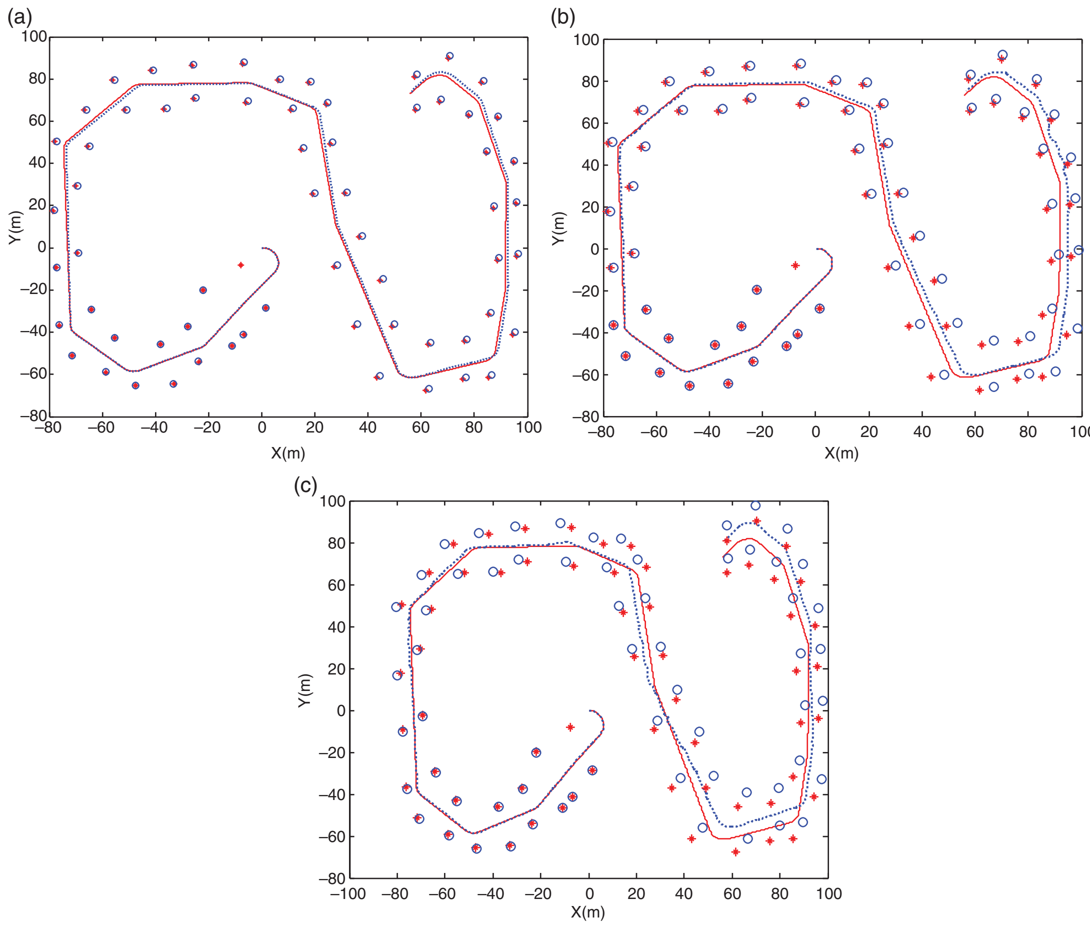

Fig. 15. Performance of algorithms: (a) Proposed method; (b) UFastSLAM; (c) FastSLAM2.0.

4.1.7. Influence of environment on the performance. To evaluate the influence size of environment on the performance of the proposed method, it is compared with other algorithms in larger environment. The measurement noise is $\sigma_r = 0.1$, $\sigma_\theta = 1$ and the control noise is $\sigma_\nu = 0.2$, $\sigma_\gamma = 2$. Figures 15 and 16 show the comparison between algorithms. It can be seen that the proposed method outperforms other methods in this environment as previous environment. Therefore, the performance of the proposed method is better than that of other algorithms in any environment.

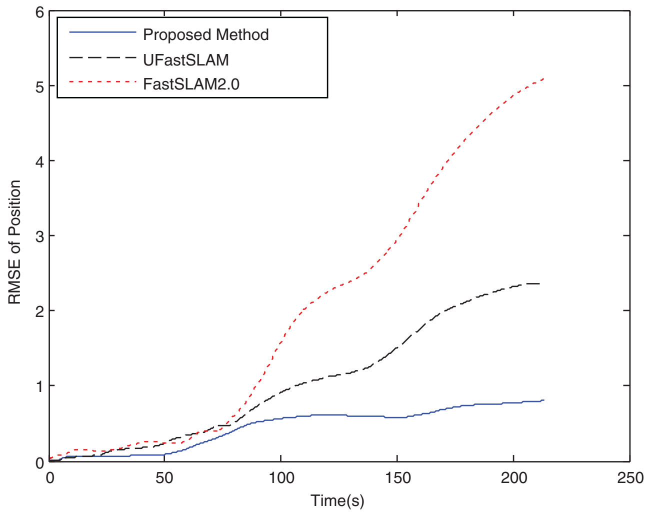

Fig. 16. RMSE of pose with respect to time.

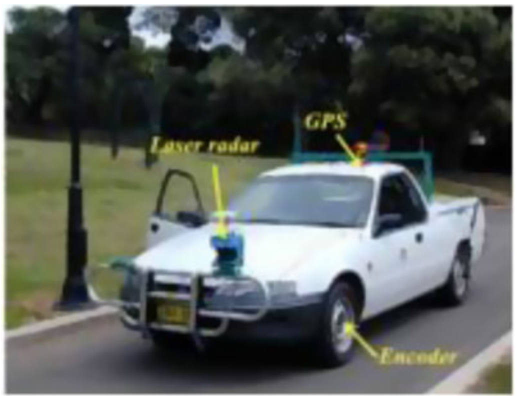

Fig. 17. Basic parameters of the mobile vehicle.[4]

4.2. Experimental

To further examine the performance of the proposed method, Car Park dataset and Sydney Victoria Park dataset[32] are utilized to validate the effectiveness of the proposed method. These datasets are a very challenging benchmark for evaluating SLAM algorithms in real-world environment. In the experiments, the performance of the proposed method is compared with that of FastSLAM2.0 and UFastSLAM. The experimental platform is a four-wheeled vehicle equipped with a GPS, a laser sensor, and wheel encoders as shown in Fig. 17.

In the Car Park test, artificial landmarks are used that consisted of 60 mm steel poles covered with reflective tape. Since the true position of the landmarks is obtained with GPS, a true map is available. In addition, a GPS receiver is used to provide ground truth for the robot position at some parts of the map. When the vehicle was driven around the park, the steering angle and the velocity were measured using encoders.

The performance of the proposed algorithm is compared with FatsSLAM2.0 and UFastSLAM in situation that the correspondences between the observation and the landmarks are assumed to be unknown and 20 particles were used for all algorithms.

Each algorithm has been executed 20 times to confirm the variance of the estimate error. For the unknown data association, the individual compatibility nearest neighbor test is used. Figure 18 shows the trajectory and landmark estimates produced by algorithms.

The result shows that in the proposed method estimated path coincides very well with the GPS path, and the estimated landmarks match the true beacons. Moreover, as shown in Fig. 19, the RMSE of the robot position in the proposed method is better than that of UFastSLAM and FastSLAM2.0.

Figure 20 shows Victoria Park with the GPS path. GPS data were collected to provide ground truth data. As shown a vehicle is driven around the park for about half an hour to traverse a trajectory

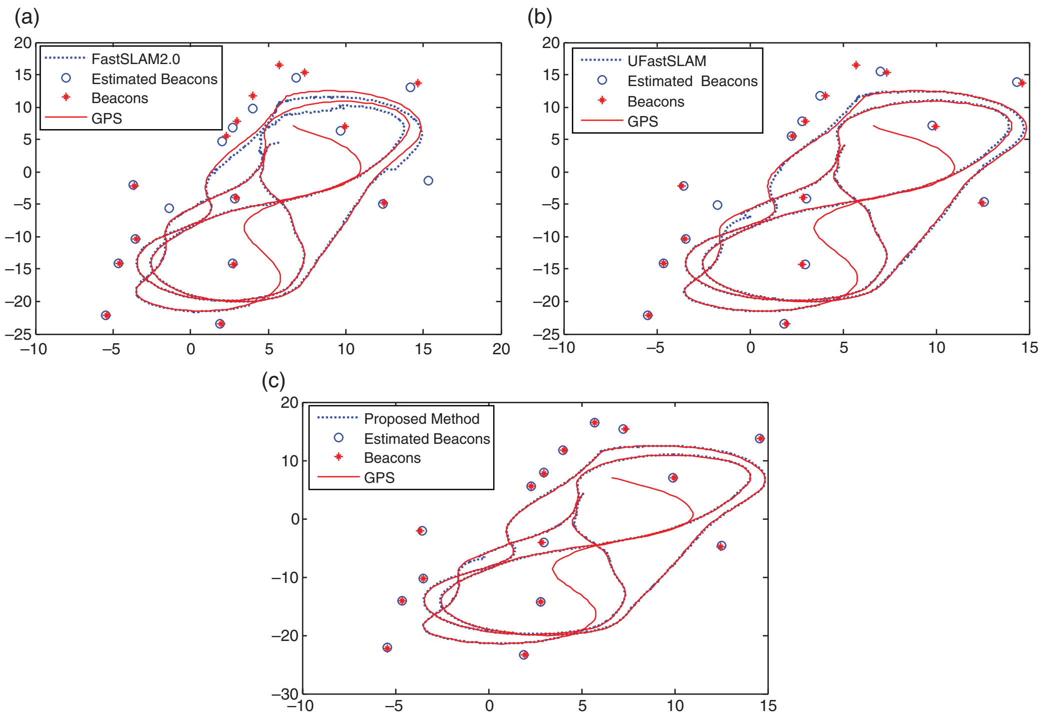

Fig. 18. (a) FastSLAM2.0; (b) UFastSLAM; (c) proposed method.

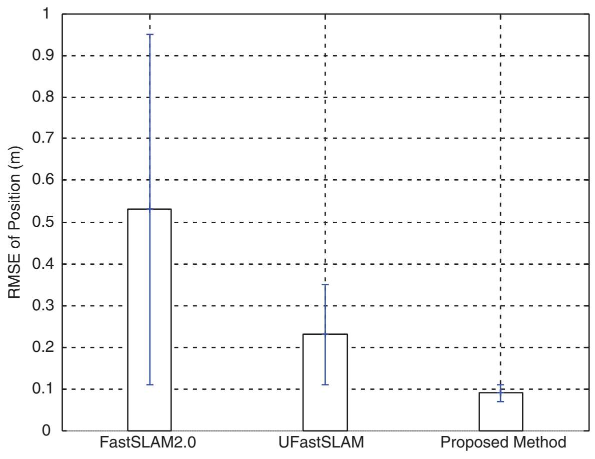

Fig. 19. RMSE of position.

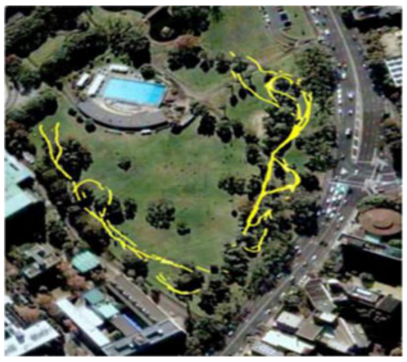

Fig. 20. Victoria Park dataset with GPS path.[4]

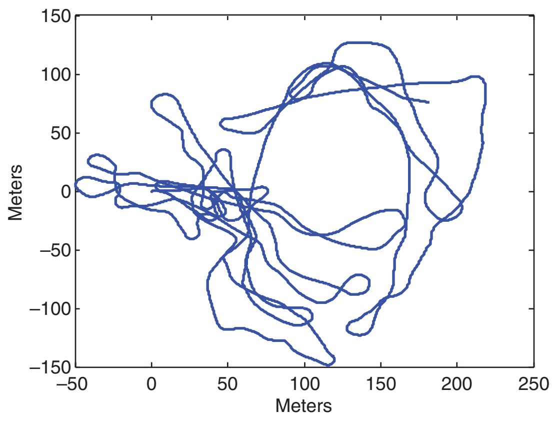

Fig. 21. Trajectory based on odometry only.

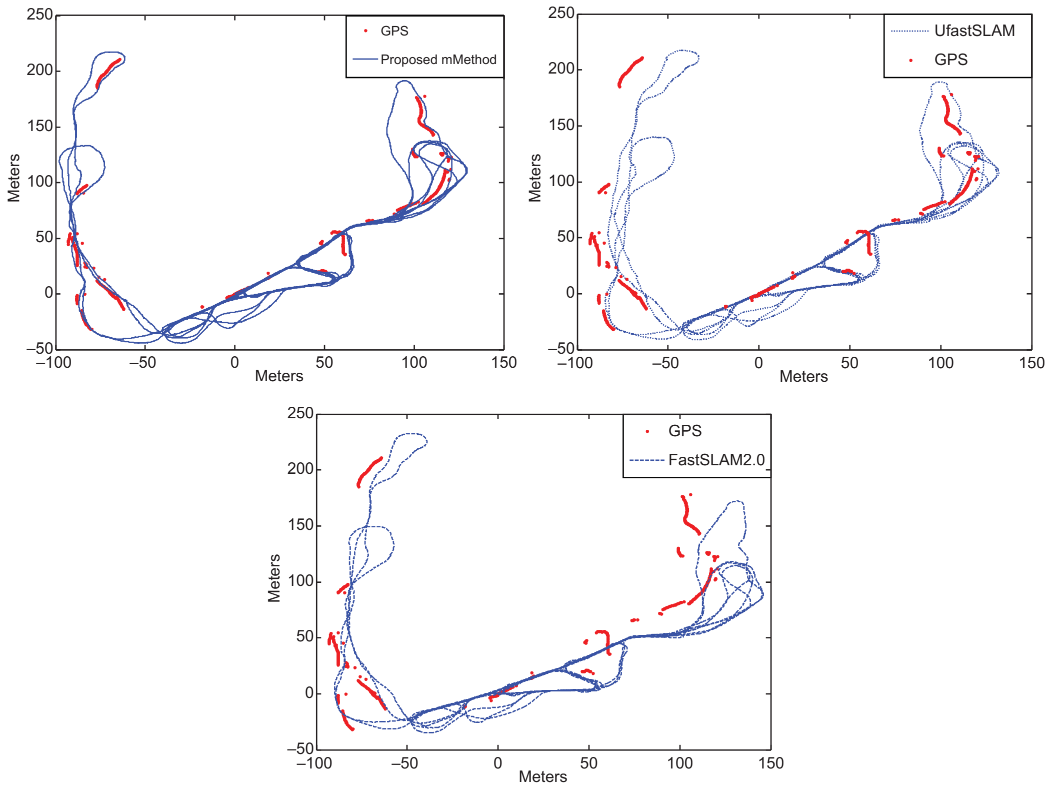

Fig. 22. Comparison of methods.

over 4 km. Since the most common features in the environment are trees, the profiles of trees are extracted from the laser data, and the centers of the trunks are then used as the point landmarks. The control inputs and feature measurements are collected from the wheel encoders and a laser range finder, respectively. Wheel encoders are used to provide odometry measurements. The GPS is provided as ground truth to evaluate the accuracy of the vehicle's estimated paths obtained by different algorithms. The numbers of particles are 30 particles. Also, the control noises in the experiment are $\sigma_v = 0.8\text{m/s}$, $\sigma_\gamma = 1.8^o$ and the measurement noises are $\sigma_r = 1.5\text{m}$, $\sigma_\phi = 2.8^o$. Each algorithm is run independently 20 times to verify the variance and the estimated error and the comparison results of accuracy for each approach. Figure 21 shows the trajectory based on odometry only. Since the robot is driving over uneven terrain, the odometry information from the encoder is poor.

Figure 22 presents the trajectories for proposed approach, Fast|SLAM2.0 and UFastSLAM. The results show that the trajectory estimated by the proposed algorithm has a better match with the true GPS points. The proposed algorithm achieves better estimation accuracy because that both the particle resampling method the calculation of the proposal distribution and are improved.

5. Conclusion

An improved FastSLAM based on RSRCKF with partial genetic resampling is proposed in this paper. In the proposed method, RSRCKF is used to design proposal distribution of the particle filter and to estimate environment landmarks. Compared to other UFastSLAM methods that require an accurate perfect knowledge of the noise statistics, the proposed method does not require a priori knowledge about the noise statistics. In addition, to increase diversity, it uses the genetic operators-based strategy to further improve the particle diversity. We conclude from the results that the proposed algorithm, even with fewer particles and unknown a priori, yields significantly more accurate and robust estimation results compared with other methods. In addition, the consistency of the proposed method is better than that of other methods.

References

1. H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: Part I the essential algorithms," *IEEE Robot. Automat. Magaz.* **13**(2), 99–110 (2006).
2. S. Huang and G. Dissanayake, "Convergence and consistency analysis for extended Kalman filter based SLAM," *IEEE Trans. Robot.* **23**(5), 1036–1049 (2007).
3. Z. Kurt Yavuz and S. Yavuz, "A Comparison of EKF, UKF, FastSLAM2.0, and UKF-based FastSLAM Algorithms," *IEEE 16th International Conference on Intelligent Engineering Systems*, Lisbon, Portugal (2012).
4. S. Thrun, M. Montemerlo, D. Koller, B. Wegbreit, J. Nieto and E. Nebot, "FastSLAM: An efficient solution to the simultaneous localization and mapping problem with unknown data association," *J. Mach. Learn. Res.* **4**(3), 380–407 (2004).
5. D. Liu, J. Duan and H. Shi, "A strong tracking square root central difference FastSLAM for unmanned intelligent vehicle with adaptive partial systematic resampling," *IEEE Trans. Intell. Transp. Syst.* **17**(1), 3110–3120 (2016).
6. C. Kim, H. Kim and W. K. Chung, "Exactly Rao-Blackwellized unscented particle filters for SLAM," *IEEE International Conference on Robotics and Automation*, Shanghai, China (2011) pp. 3589–3594.
7. C. Kim, R. Sakthivel and W. K. Chung, "Unscented FastSLAM: A robust and efficient solution to the SLAM problem," *IEEE Trans. Robot.* **24**(4), 808–820 (2008).
8. 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. Automat. Contr.* **45**(3), 477–482 (2000).
9. R. K. Mehra, "On the identification of variances and adaptive Kalman filtering," *IEEE Trans. Automat. Contr.* **AC-15**(2), 175–184 (1970).
10. R. J. Fitzgerald, "Divergence of the Kalman filter," *IEEE Trans. Automat. Contr.* **AC-16**(6), 736–747 (1971).
11. Y. Shi, C. Han and Y. Liang, "Adaptive UKF for target tracking with unknown process noise statistics," *The 12th International Conference on Information Fusion*, Seattle, WA, July 6–9 (2009).
12. Z. Jiang, Q. Song, Y. He and J. Han, "A novel adaptive unscented Kalman filter for nonlinear estimation," *The 46th IEEE Conference on Decision and Control*, New Orleans, LA, December 12–14 (2007).
13. R. Havangi, M. Teshnelab, M. A. Nekoui and H. Taghirad, "An adaptive Neuro-Fuzzy Rao-Blackwellized particle filter for SLAM," *Proceedings of the IEEE International Conference on Mechatronics*, Istanbul, Turkey (2011) pp. 487–492.
14. Z. Qiu, H. Qian and G. Wang, "Adaptive robust cubature Kalman filtering for satellite attitude estimation", *Chinese J. Aeronaut.* **31**(4), 806–819 (2017).
15. I. Rasaratnam, S. Haykin and T. R. Hurd, "Cubature Kalman filtering for continuous-discrete systems: Theory and simulations," *IEEE Trans. Signal Process.* **58**(10), 4977–4993 (2010).
16. N. Kwak, G.W. Kim and B.H. Lee, "A new compensation technique based on analysis of resampling process in FastSLAM," *Robotica* **26**(2), 205–217 (2008).
17. M. Bolic, P. M. Djuric and S. Hong, "Resampling algorithms and architectures for distributed particle filters," *IEEE Trans. Signal Process.* **53**(7), 442–2450 (2005).
18. C. Fen, M. Wang and Q. B. Ji, "Analysis and comparison of resampling algorithms in particle filter," *J. Syst. Simul.* **21**(4), 1101–1105 (2009).
19. R. Havangi, "Intelligent FastSLAM: An intelligent factorized solution to simultaneous localization and mapping," *Int. J. Hum. Robot*. **14**(1), 1650026-1–1650026-20 (2017).
20. D. Jian-min, L. Dan, Y. Hong-Xiao, and S. Hui, "An improved FastSLAM algorithm for autonomous vehicle based on the strong tracking square root central difference Kalman filter," *Proceedings of ITSC* (2015), pp. 693–698.
21. A. R. Khairuddin and M. S. Talib, "GA-PSO-FASTSLAM: A hybrid optimization approach in improving FastSLAM performance," *International Conference on Intelligent Systems Design and Applications*, Vellore, India (2017).

22. T. Lv and M. Feng, "An improved FastSLAM 2.0 algorithm based on FC&ASD-PSO," *Robotica* **35**(9), 1795–1815 (2017).
23. B. He, L. Ying, S. Zhang, X. Feng, T. Yan, R. Nian and Y. Shen, "Autonomous navigation based on unscented-FastSLAM using particle swarm optimization for autonomous underwater vehicles," *Measurement* **71**, 89–101 (2015).
24. S. wan Lee, G. Eoh and B. H. Lee, "Relational FastSLAM: An improved Rao-Blackwellized particle filtering framework using particle swarm characteristics," *Robotica* **34**(6), 1282–1296 (2016).
25. I. Arasaratnam and S. Haykin, "Cubature Kalman filters," *IEEE Trans. Automat. Contr.* **54**(6), 1254–1269 (2009).
26. I. Arasaratnam, S. Haykin and R. T. Hurd, "Cubature Kalman filtering for continuous-discrete system: Theory and simulations," *IEEE Trans. Signal Process.* **58**(10), 4977–4993 (2010).
27. Y. Huang and Y. Zhang, "Robust student's t-based stochastic cubature filter for nonlinear systems with heavy-tailed process and measurement noises," *IEEE Access* **5**, 7964–7974 (2017).
28. K. Nishiyama, "Computational improvement of the fast $H\infty$ filter based on information of input predictor," *IEEE Trans. Signal Process.* **55**(8), 4316–4320 (2007).
29. K. Nishiyama, "An $H\infty$ optimization and its fast algorithm for time-variant system identification," *IEEE Trans. Signal Process.* **52**(5), 1335–1342 (2004).
30. M. S. Arulampalam, S. Maskell, N. Gordon and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Trans. Signal Process.* **50**(2), 174–188 (2002).
31. Y. Bar-Shalom, X. R. Li and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation* (John Wiley & Sons, 2001).
32. E. Nebot, "Victoria Park Data Set," http://www-personal.acfr.usyd.edu.au/nebot/dataset.htm (2012).