Received July 22, 2019, accepted August 9, 2019, date of publication August 13, 2019, date of current version August 28, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2934995

Intelligent Filter-Based SLAM for Mobile Robots With Improved Localization Performance

MINGWEI LIN[1,2], (Member, IEEE), CANJUN YANG[1,2], DEJUN LI[1,2], AND GENGLI ZHOU[1]
[1]State Key Laboratory of Fluid Power and Mechatronic Systems, Zhejiang University, Hangzhou 310027, China
[2]Pilot National Laboratory for Marine Science and Technology, Qingdao 266000, China

Corresponding authors: Mingwei Lin (lmw@zju.edu.cn) and Canjun Yang (ycj@zju.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 41676089, and in part by the Marine S&T Fund of Shandong Province under Grant 2018SDKJ0211.

ABSTRACT Fast simultaneous localization and mapping (FastSLAM) is one of the most popular methods for autonomous navigation of mobile robots. However, FastSLAM is essentially a particle filter (PF) that suffers from particle impoverishment and degeneracy problems. To improve its localization performance, this paper proposes an improved FastSLAM algorithm that contains an intelligent bat-inspired resampling whose iteration times can be adaptively tuned based on the degree of filter diverging. Additionally, the square root cubature filter is merged into the algorithm for better proposal distribution and mapping results. The advantages of the proposed method are verified by simulation and dataset-based tests. The test result demonstrates that the proposed IFastSLAM has better accuracy, computational efficiency and filter consistency compared to that of the square root unscented FastSLAM (SRUFastSLAM) and strong tracking square root central difference FastSLAM (STSRCDFastSLAM). Finally, a pool experiment is demonstrated to further verify the advantages of the proposed algorithm.

INDEX TERMS Particle filter, adaptive bat-inspired resampling, filter-based SLAM, mobile robots.

I. INTRODUCTION

Simultaneous localization and mapping (SLAM), also referred to as concurrent mapping and localization (CML), is a popular method to improve localization accuracy of mobile robots. By fusing different sensor data, the uncertainty of the robot pose can be significantly reduced in SLAM. One of the most popular SLAM algorithms is the extended Kalman filter SLAM (EKF-SLAM) [1], [2]. Conventional EKF-SLAM has the filter inconsistency problem caused by the hypothesis of the Gaussian noise type and the linearization process [3]. Moreover, the computational burdens of computing Jacobian matrixes with the growing map size also restricts the algorithm to large-scale environments.

To overcome these defects, a number of particle filter-based SLAM algorithms have been proposed [4]–[6]. A typical variant of those algorithms is Rao-Blackwellised particle filter based SLAM, which is extensively used in SLAM community and known as FastSLAM [7]. The major advantage of FastSLAM is that it can factorize the full posterior distribution into a product of landmark distributions and a vehicle path distribution, thus reducing the computational burdens of the algorithm. Moreover, by representing the vehicle pose with samples (particles), the FastSLAM can address the non-linearity issues and the accumulated error caused by assumption of Gaussian noise. So far, many FastSLAM variants have shown particular advantages in real-world dataset test or experiment [8]–[10]. In general, FastSLAM contains two essential operations: 1) importance sampling, 2) resampling. There have been various representative non-linear filters serving as the importance sampling functions, such as the unscented Kalman filter (UKF) [11] and cubature Kalman filter (CKF) [12]. However, the inherent drawbacks of the filters and limited number of particles would cause severe degeneration of FastSLAM. Therefore, a resampling step is generally followed by the importance sampling step in FastSLAM to improve the algorithm performance [13]. The common resampling methods, such as the systematic resampling, multinomial resampling and the residual resampling, have proved to be well-performed in preventing deterioration of particle filters. However, their redundant particles would lower diversity of the particles and thus cause imprecise approximation of the robot state.

The associate editor coordinating the review of this article and approving it for publication was Haiquan Zhao.

To overcome the above-mentioned problem, this paper proposes an intelligent resampling step to FastSLAM, which is inspired by microbat behaviors to obtain the optimized state of the robot [14]. Moreover, the proposed resampling only operates on small-weight particles and thus reduce the computational burden. In the proposed FastSLAM algorithm, the square root cubature Kalman filter (SRCKF) is fused for stepwise update of robot poses and map features [15], instead of updating in an augmented way [16]. In this paper, the improved FastSLAM is referred to as IFastSLAM. The proposed algorithm is verified by comparing to the square root unscented FastSLAM (SRUFastSLAM) and the strong tracking square root central difference FastSLAM (STSRCD-FastSLAM) in simulation and dataset tests [17], [18]. Finally, a pool experiment is demonstrated to further validate the performance of IFastSLAM.

The rest of this paper is organized as follows. Section II introduces the robot state and measurement functions. Section III introduces the proposed IFastSLAM algorithm with adaptive bat-inspired resampling. Section IV and V demonstrate the numerical test and experiment using the simulator and datasets collected in Car Park and Victoria Park, respectively [19], [20]. Section VI demonstrates the pool experiment, and section VII draws a conclusion and discusses the future work.

II. ROBOT STATE AND MEASUREMENT FUNCTIONS

This paper studies on the horizontal location of the robot and uses a three-dimensional vector to describe the robot pose. The i-th cubature point regarding the robot state is

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

where $(\chi_{x,t-1}^{[i][m]}\ \chi_{y,t-1}^{[i][m]})^T$ and $\chi_{\theta,t-1}^{[i][m]}$ denote the robot position and azimuth angle of the i-th cubature point, respectively. The control inputs, velocity V_t and steering angle G_t of the robot, with the added control noise component $\chi_t^{u[i][m]}$, are demonstrated as follows

$$\begin{bmatrix} V_n \\ G_n \end{bmatrix} = \begin{bmatrix} V_t + \chi_{V,t}^{u[i][m]} \\ G_t + \chi_{G,t}^{u[i][m]} \end{bmatrix} \tag{2}$$

where

$$\chi_t^{u[i][m]} = \begin{bmatrix} \chi_{V,t}^{u[i][m]} \\ \chi_{G,t}^{u[i][m]} \end{bmatrix} \tag{3}$$

Here, the Ackerman model is the process state function, and it is used for four-wheel robots. The propagation of the cubature points of the state $\chi_{t-1}^{[i][m]}$ is (4), as shown at the top of the next page, where L is the wheel base and Δt is the sample interval. The geometry of parameter a and b is shown in Fig.1.

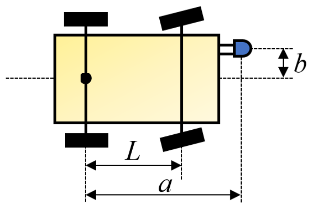


FIGURE 1. Parameters of the four-wheel robot.

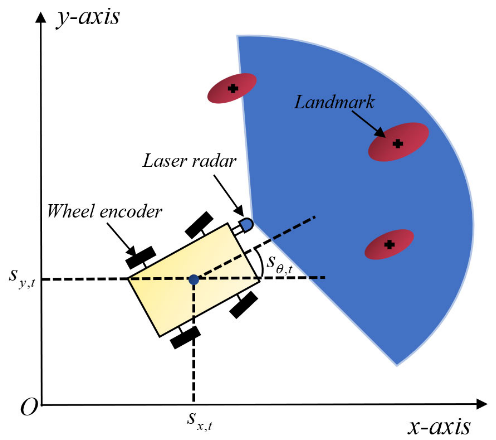


FIGURE 2. Localization schematic diagram of the four-wheel robot.

The measurement function of the robot is given by

$$z_{j,t} = \begin{bmatrix} \rho \\ \theta \end{bmatrix} = \begin{bmatrix} \sqrt{(s_{x,t} - x_j)^2 + (s_{y,t} - y_j)^2} \\ \arctan(\frac{s_{y,t}-y_j}{s_{x,t}-x_j}) - s_{\theta,t} + \frac{\pi}{2} \end{bmatrix} \tag{5}$$

where $z_{j,t}$ is the measurement vector of the j-th feature in the polar coordinate, (x_j, y_j) is the j-th observed feature in the Cartesian coordinate, and $(s_{x,t}, s_{y,t}, s_{\theta,t})$ is the robot state at time step t.

Fig.2 demonstrates the schematic diagram of a four-wheel robot, which is equipped with a laser radar and rear wheel encoders. The laser radar can detect the environmental landmarks and the wheel encoder can provide the velocity and steering angle of the robot. It's noted that the proposed IFastSLAM is not restricted to the system model or equipped sensors. For example, the laser radar can be changed to cameras or underwater sonars for underwater environment, and the wheel encoder can be changed to the inertia measurement unit (IMU) or other sensor suite which can provide the velocity and angle information as long as the model and sensor information can match with each other.

III. IFASTSLAM

This section introduces the fundamental theory of FastSLAM, update process of robot and feature states, adaptive bat-inspired resampling, and optimization process of particle distribution,
respectively.

$$\chi_t^{[i][m]} = \chi_{t-1}^{[i][m]} + \begin{bmatrix} \Delta t \left\{ V_n \cos(\chi_{\theta,t-1}^{[i][m]}) - \dfrac{V_n \tan(G_n)}{L}(a \sin(\chi_{\theta,t-1}^{[i][m]}) + b\cos(\chi_{\theta,t-1}^{[i][m]})) \right\} \\ \Delta t \left\{ V_n \sin(\chi_{\theta,t-1}^{[i][m]}) - \dfrac{V_n \tan(G_n)}{L}(a \cos(\chi_{\theta,t-1}^{[i][m]}) - b\sin(\chi_{\theta,t-1}^{[i][m]})) \right\} \\ V_n \Delta t \tan(G_n)/L \end{bmatrix} \tag{4}$$

A. FUNDAMENTAL THEORY OF FASTSLAM

Generally, the posterior probability of the robot trajectory and the map can be represented from a probabilistic view [7]

$$p(s^t, \theta \mid z^t, u^t, n^t) \tag{6}$$

where the trajectory of the vehicle is given by $s^t = \{s_1, \ldots, s_t\}$, and θ represents the map. Each landmark is denoted by θ_k for $k = 1, \ldots, N$ where N is the number of stationary landmarks perceived by the vehicle. $z^t = \{z_1, \ldots, z_t\}$ and $u^t = \{u_1, \ldots, u_t\}$ are the measurements and controls/odometry information until time t, respectively. $n^t = \{n_1, \ldots, n_t\}$ denotes the data association information where n_t determines the identity of the landmark observed at time t. The factorization of the posterior probability is demonstrated as follows [7]

$$\underbrace{p(s^t, \theta | z^t, u^t, n^t)}_{SLAM\ posterior} = \underbrace{p(s^t | z^t, u^t, n^t)}_{Trajectory\ posterior} \underbrace{\prod_{k=1}^{K} p(\theta_k | s^t, z^t, u^t, n^t)}_{Feature\ posterior} \tag{7}$$

Each particle m is given by:

$$S_t^{[m]} = \{s^{t,[m]}, \mu_{1,t}^{[m]}, \sum\nolimits_{1,t}^{[m]}, \ldots, \mu_{K,t}^{[m]}, \sum\nolimits_{K,t}^{[m]}\} \tag{8}$$

where $[m]$ is the index of the particle, $s^{t,[m]}$ is the estimated path of the$m$-th particle, $\mu_{K,t}^{[m]}$ and $\sum_{K,t}^{[m]}$ are the mean and the covariance of the Gaussian distribution of the K-th feature location updated by m-th particle. Hence, the robot pose s_t is sampled by

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

The importance weight $w_t^{[m]}$ of each particle is given by [21], [22]

$$w_t^{[m]} = \frac{\text{target distribution}}{\text{proposal distribution}} = \frac{p(s^{t,[m]} \big| z^t, u^t, n^t)}{p(s^{t-1,[m]} \big| z^{t-1}, u^{t-1}, n^{t-1}) p(s_t^{[m]} \big| s^{t-1,[m]} z^t, u^t, n^t)} \tag{10}$$

B. ESTIMATION OF THE ROBOT STATE

To improve the localization performance of FastSLAM, the SRCKF is used to generate the proposal distribution [15]. Firstly, the state vector is augmented by adding the mean and covariance of the process noise. Assuming that the mean of the process noise is zero, the augmented state is formulated as

$$s_{t-1}^{a[m]} = \begin{bmatrix} s_{t-1}^{[m]} \\ 0 \end{bmatrix}, \quad s_{t-1}^{[m]} = \begin{bmatrix} s_{x,t-1}^{[m]} \\ s_{y,t-1}^{[m]} \\ s_{\theta,t-1}^{[m]} \end{bmatrix} \tag{11}$$

$$\mathbf{P}_{t-1}^{a[m]} = \begin{bmatrix} \mathbf{P}_{t-1}^{[m]} & 0 \\ 0 & \mathbf{Q}_{t-1} \end{bmatrix} \tag{12}$$

where $s_{t-1}^{a[m]}$ denotes the augmented vector, $s_{t-1}^{[m]}$ and $\mathbf{P}_{t-1}^{[m]}$ denote the mean and covariance of the robot at the last time step $t-1$, and $(s_{x,t-1}^{[m]}, s_{y,t-1}^{[m]}, s_{\theta,t-1}^{[m]})^T$ denotes the state vector of the robot. The dimension of $s_{t-1}^{a[m]}$ is n_a. Applying the Cholesky factorization to decompose the matrix $P_{t-1}^{a[m]}$, it can be calculated as

$$\mathbf{M}_{t-1}^{a[m]} = chol\left(\mathbf{P}_{t-1}^{a[m]}\right) \tag{13}$$

where the operator $chol(\cdot)$ denotes the Cholesky factorization which decomposes a positive-definite matrix $\mathbf{P}_{t-1}^{a[m]}$ into the product of $(\mathbf{M}_{t-1}^{a[m]})^T \mathbf{M}_{t-1}^{a[m]}$, and $\mathbf{M}_{t-1}^{a[m]}$ denotes its upper triangular matrix. The set of $2n_a$ cubature points are calculated as

$$\chi_{t-1}^{a[i][m]} = s_{t-1}^{a[m]} + (\mathbf{M}_{t-1}^{a[m]})^T \zeta^{[i]}, \quad i = 1, 2, \ldots, 2n_a \tag{14}$$

where

$$\zeta = \sqrt{n_a} \left\{ \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{pmatrix}, \ldots, \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{pmatrix}, \begin{pmatrix} -1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ -1 \\ \vdots \\ 0 \end{pmatrix}, \ldots, \begin{pmatrix} 0 \\ 0 \\ \vdots \\ -1 \end{pmatrix} \right\}_{n_a \times 2n_a} \tag{15}$$

and $\zeta^{[i]}$ is the i-th column vector of (15). Each $\chi_{t-1}^{a[i][m]}$ contains the state and process noise components that given by

$$\chi_{t-1}^{a[i][m]} = \begin{bmatrix} \chi_{t-1}^{[i][m]} \\ \chi_t^{u[i][m]} \end{bmatrix} \tag{16}$$

The set of cubature points are propagated through the nonlinear process function as (4):

$$\bar{\chi}_{t|t-1}^{[i][m]} = f(u_t + \chi_t^{u[i][m]}, \chi_{t-1}^{[i][m]}) \tag{17}$$

where $\bar{\chi}_{t|t-1}^{[i][m]}$ is the propagated cubature points of the robot state. The predicted state is calculated based on the weights as follows:

$$s_{t|t-1}^{[m]} = \sum_{i=1}^{2n_a} W \bar{\chi}_{t|t-1}^{[i][m]} \tag{18}$$

where $W = 1/(2n_a)$. It is computationally expensive to carry out the Cholesky factorization because it contains a set of weighted deviation e_i as shown in (19). Hence, the QR decomposition is performed on the matrix of $A = [e_1 e_2 \ldots e_{2n_a}]$ to reduce the computational efforts as [23]

$$e_i = \sqrt{W}\left(\bar{\chi}_t^{[i][m]} - s_{t|t-1}^{[m]}\right), \quad i = 1, \ldots, 2n_a \tag{19}$$

$$\bar{\mathbf{M}}_{t|t-1}^{[m]} = qr(A) = qr[e_1\ e_2 \ldots e_{2n_a}] \tag{20}$$

where $\bar{\mathbf{M}}_{t|t-1}^{[m]}$ is the predicted factor. The treatment of the measurement is demonstrated in a similar way. If some landmarks are observed, the data association will provide their identities. Therefore, the predicted measurement $\hat{z}_t^{[m]}$ is calculated as follows

$$\begin{aligned} \bar{\gamma}_t^{[i][m]} &= h(\bar{\chi}_{t|t-1}^{[i][m]}, \mu_{t-1}^{[m]}) \\ \hat{z}_t^{[m]} &= \sum_{i=1}^{2n_a} W \bar{\gamma}_t^{[i][m]} \end{aligned} \tag{21}$$

where $\bar{\gamma}_t^{[i][m]}$ is the propagated measurement by the SRCKF where $h(\cdot)$ denotes the measurement function as (5). The weighted and centered matrix is

$$\Psi_{z_t}^{[m]} = \frac{1}{\sqrt{W}}[\varsigma_1\ \varsigma_2\ \ldots\ \varsigma_{2n_a}] \tag{22}$$

where

$$\varsigma_i = \bar{\gamma}_t^{[i][m]} - \hat{z}_t^{[m]}, \quad i = 1, \ldots, 2n_a \tag{23}$$

The square root of the innovation covariance matrix is thus given by

$$\bar{\mathbf{S}}_{z_t}^{[m]} = qr[\Psi_{z_t}^{[m]}\ \sqrt{\mathbf{R}_{t-1}}] \tag{24}$$

where $\mathbf{R}_{t-1}$ is the measurement noise covariance. $\bar{\mathbf{S}}_{z_t}^{[m]}$ is the upper triangular matrix of the covariance matrix of $\hat{z}_t^{[m]}$. The cross covariance of the state is given by

$$\mathbf{P}_{s_t z_t}^{[m]} = \beta_{s_t}^{[m]}\left(\Psi_{z_t}^{[m]}\right)^T \tag{25}$$

where

$$\beta_{s_t}^{[m]} = \frac{1}{\sqrt{W}}\big[\bar{\chi}_t^{[i][m]} - s_{t|t-1}^{[m]},\ \bar{\chi}_t^{[i][m]} - s_{t|t-1}^{[m]}, \ldots, \bar{\chi}_t^{[2n_a][m]} - s_{t|t-1}^{[m]}\big].$$

The Kalman gain matrix and measurement update are given by

$$\mathbf{K}_t^{[m]} = \left(\mathbf{P}_{s_t z_t}^{[m]}/\bar{\mathbf{S}}_{z_t}^{[m]}\right)/\left(\bar{\mathbf{S}}_{z_t}^{[m]}\right)^T \tag{26}$$

$$\hat{s}_t^{[m]} = s_{t|t-1}^{[m]} + \mathbf{K}_t^{[m]}(z_t - \hat{z}_t^{[m]}) \tag{27}$$

$$\mathbf{M}_t^{[m]} = qr(\beta_{s_t}^{[m]} - \mathbf{K}_t^{[m]}\bar{\mathbf{S}}_{z_t}^{[m]},\ \mathbf{K}_t^{[m]}\sqrt{\mathbf{R}_{t-1}}) \tag{28}$$

where z_t is the observation input, $s_t^{[m]}$ is the state mean, and $\mathbf{M}_t^{[m]}$ is the upper triangular decomposition of the covariance matrix. For the Gaussian distribution, the state of each particle is sampled by

$$s_t^{[m]} \sim N(\hat{s}_t^{[m]}, \left(\mathbf{M}_t^{[m]}\right)^T \mathbf{M}_t^{[m]}). \tag{29}$$

Eqs.(31) is known as the importance sampling step. The robot state can be calculated by the method of weighted mean, and the weight of each particle is calculated as [21]

$$w_t^{[m]} = w_{t-1}^{[m]} \frac{p(z_t|\hat{s}_t^{[m]})p(s_t^{[m]}|s_{t-1}^{[m]}, u_t)}{N(\hat{s}_t^{[m]}, P_t^{[m]})}. \tag{30}$$

Normalize the weight of each particle as

$$\hat{w}_t^{[m]} = w_t^{[m]} / \sum_{i=1}^{M} w_t^{[i]} \tag{31}$$

where $\hat{w}_t^{[m]}$ denotes the normalized weight of m-th particle and M denotes the particle number. Base on the previous calculation, the robot state can be updated as

$$s_t = \sum_{i=1}^{M} \hat{w}_t^{[m]} s_t^{[m]}. \tag{32}$$

C. ESTIMATION OF MAP FEATURES

The feature estimate can be divided into two cases. One case is that the robot revisits the landmark stored in the map, and the other case is that the robot observes new features, which should be added to the map.

1) UPDATE REVISITED LANDMARKS

The SRCKF is used to estimate the feature location here. If a landmark n_t is revisited at time t, the cubature points of the feature are defined using the previously registered mean $\mu_{n_t,t-1}^{[m]}$ as follows

$$\delta^{[i][m]} = \mu_{n_t,t-1}^{[m]} + \Omega_{n_t,t-1}^{[m]}\lambda^{[i]}, \quad i = 1, \ldots, 2n \tag{33}$$

$$\lambda = \sqrt{n}\left\{\begin{pmatrix}1\\0\\\vdots\\0\end{pmatrix}, \begin{pmatrix}0\\1\\\vdots\\0\end{pmatrix}, \ldots, \begin{pmatrix}0\\0\\\vdots\\1\end{pmatrix}, \begin{pmatrix}-1\\0\\\vdots\\0\end{pmatrix}, \begin{pmatrix}0\\-1\\\vdots\\0\end{pmatrix}, \ldots, \begin{pmatrix}0\\0\\\vdots\\-1\end{pmatrix}\right\}_{n\times 2n} \tag{34}$$

where $\Omega_{n_t,t-1}^{[m]}$ is the square root of the feature covariance matrix, and n is the dimension of the feature state. If the environmental landmarks are all located on one plane, n can be set at 2. The cubature points are propagated through the measurement model as follows

$$\bar{Z}_t^{[i][m]} = h(\delta^{[i][m]}, s_t^{[m]}) \tag{35}$$

$$\hat{\Gamma}_t^{[m]} = \sum_{i=1}^{2n} W_n \bar{Z}_t^{[i][m]}. \tag{36}$$

The weighted matrixes for the measurement and feature state are defined as

$$\Xi_{z,t} = \sqrt{W_n}\left[\bar{Z}_t^{[1][m]} - \hat{\Gamma}_t^{[m]}, \ldots, \bar{Z}_t^{[2n][m]} - \hat{\Gamma}_t^{[m]}\right] \tag{37}$$

$$\Xi_{\mu,t} = \sqrt{W_n}\left[\delta^{[1][m]} - \mu_{n_t,t-1}^{[m]}, \ldots, \delta^{[2n][m]} - \mu_{n_t,t-1}^{[m]}\right]. \tag{38}$$

The square root factor of the innovation covariance and the cross covariance are calculated as

$$\bar{\mathbf{S}}_{\Gamma_t}^{[m]} = qr\left(\Xi_{z,t}, \sqrt{\mathbf{R}_t}\right) \tag{39}$$

$$\bar{\Lambda}_{z_t s_t}^{[m]} = \Xi_{\mu,t}\left(\Xi_{z,t}\right)^T \tag{40}$$

where $W_n = 1/(2n)$. The Kalman gain is calculated as

$$\bar{\mathbf{K}}_t^{[m]} = \left(\bar{\Lambda}_{z_t s_t}^{[m]} / \bar{\mathbf{S}}_{\Gamma_t}^{[m]}\right) / \left(\bar{\mathbf{S}}_{\Gamma_t}^{[m]}\right)^T. \tag{41}$$

Finally, the mean and the square root factor of the feature covariance are updated as

$$\mu_{n_t,t}^{[m]} = \mu_{n_t,t-1}^{[m]} + \bar{\mathbf{K}}_t^{[m]}(z_t - \hat{\Gamma}_t^{[m]}) \tag{42}$$

$$\sum_t^{[m]} = qr\left(\Xi_{\mu,t} - \bar{\mathbf{K}}_t^{[m]}\Xi_{z,t}, \bar{K}_t^{[m]}\sqrt{\mathbf{R}_t}\right). \tag{43}$$

2) INITIALIZING NEW LANDMARKS

If a new landmark is observed at the time step t, the measurement z_t of the sensor and the measurement noise covariance $\mathbf{R}_t$ are used to initialize the features whose cubature points are given by

$$\psi^{[i][m]} = z_t + \sqrt{\mathbf{R}_t}\zeta^{[i]}, \quad i = 1, \ldots, 2n. \tag{44}$$

The mean and square root factor of covariance of the newly visited feature are given by:

$$\vartheta_t^{[i][m]} = h^{-1}\left(\psi^{[i][m]}, \hat{s}_t^{[m]}\right) \tag{45}$$

$$\mu_{n_t,t}^{[m]} = \sum_{i=1}^{2n} W_n \vartheta_t^{[i][m]} \tag{46}$$

$$\tau_i = \sqrt{W_n}\left(\vartheta_t^{[i][m]} - \mu_{n_t,t}^{[m]}\right), \quad i = 1, \ldots, 2n \tag{47}$$

$$\Omega_t^{[m]} = qr[\tau_1\ \tau_2\ \ldots\ \tau_{2n}]. \tag{48}$$

D. ADAPTIVE BAT-INSPIRED RESAMPLING

Common resampling methods overcome the degeneracy problem by copying the large-weight particles to replace the small-weight particles. However, the resulting redundant particles cannot well approximate the PDF of the true state due to loss of particle diversity. Some improved methods only conduct the resampling on partial particles, in order to overcome the impoverishment problem [17], [18], while the replaced particles are empirically selected. Additionally, these approaches still contain some repeated particles and thus cause the particle degeneracy. Therefore, an adaptive resampling inspired by bat behaviors is proposed to enhance the performance of FastSLAM. Note that the bat algorithm is originally used for multi-objective optimization problems [18]. Here, it is modified as an intelligent resampling step.

1) CALCULATION OF RESAMPLING THRESHOLD

The degeneracy degree of the proposed IFastSLAM can be evaluated by the effective particle number N_{eff} which is calculated as follows [17], [24]

$$N_{eff} = \frac{1}{\sum_{m=1}^{M}(\hat{w}_t^{[m]})^2}. \tag{49}$$

If the effective particle number is smaller than the threshold, the resampling step will be carried out.

2) ADAPTIVE PARTICLE CLASSIFICATION

The particle set $\{s_t^{[m]}, \hat{w}_t^{[m]}\}$ is sorted in a descending order denoted as $\Theta = \{\hat{w}_t^{[1]}, \hat{w}_t^{[2]}, \ldots, \hat{w}_t^{[M]}\}$ where $m = 1, \ldots, M$. It's approved that a particle with higher weight is more likely to tend to the true posterior distribution compared to that with a smaller weight [25]. Therefore, the particle with the highest weight in Θ is regarded as the optimal particle and it is denoted as s_{best}. To improve the computational efficiency of the IFastSLAM, the resampling step is only conducted on the small-weight particles. An adaptive segment threshold is proposed as:

$$\alpha_{th} = ceil(N_{eff}) \tag{50}$$

where the function $ceil(\cdot)$ can return the value of a number rounded upwards to the nearest integer. If the order of a particle in Θ is larger than α_{th}, it should be resampled. The resampled particle set is denoted as Θ_{small}.

3) BAT-INSPIRED OPTIMIZATION

In the bat-inspired resampling, a bat corresponds to a particle in the PF. The algorithm includes three steps:

First, generate new solutions for global search. In this step, each bat is randomly assigned a frequency drawn from $[f_{\min}, f_{\max}]$, and it locates at position s_i with velocity v_i where i denotes the order in Θ. Driven by the globally best bat, the off-spring would tend to a better position.

Second, randomly walk around a selected best solution. This step is used to prevent the algorithm from the local minimal. Once a solution is selected as the best solution, a random walk strategy is used to extend the search space where ε is a random number vector whose components are drawn from the uniform the distribution $[-1, 1]$ and A_{avg} is the average loudness of all the bats at this time step.

Third, update the bat population. If the bat has a higher weight than that of parents, it means the robot localization is improved. Then, the bat location, pulse rate r_i and loudness A_i will be updated for the next period.

4) CALCULATION OF ITERATION TIMES

Different from the case in conventional optimization problems, the particles which need to be resampled have already located around the true state. Therefore, the resampling step doesn't need many optimization iterations. Particle filter failure/diverging sometimes occurs when the noise is small or the particle number is not adequate. The feature of filter failure is presented in the way that no samples or only a few samples of predicted measurement fall within the measurement ellipse. Here, the filter diverging degree is used to determine the BA iteration times, which is evaluated by the Mahalanobis distance D_t of the measurement [26]:

$$\bar{s}_t = \sum_{m=1}^{M} \hat{w}_t^{[m]} s_t^{[m]} \tag{51}$$

$$\bar{Z}_t = h(\bar{s}_t) \tag{52}$$

$$D_t = (z_t - \bar{Z}_t)^T R_t^{-1} (z_t - \bar{Z}_t) \tag{53}$$

where $\bar{s}_t$ denotes the mean of the posterior particles, $\bar{Z}_t$ denotes the predicted measurement. When D_t is less than the *chi-square* value χ^2, the PF can be regarded to normally work, otherwise the PF gets trapped in diverging. Based on the above-mentioned analysis, an adaptive tuning strategy is proposed to calculate the iteration times of bat optimization in the resampling step as

$$p = \begin{cases} N_{\min}, & \frac{D_t}{\chi^2} \le 1 \\ ceil\left(N_{\min} + \frac{(D_t/\chi^2 - 1)(N_{\max} - N_{\min})}{(a_1 - 1)}\right), & 1 < \frac{D_t}{\chi^2} \le a_1 \\ N_{\max}, & \frac{D_t}{\chi^2} > a_1 \end{cases} \tag{54}$$

where p denotes the iteration times. $N_{\min}$ denotes the initial iteration times, and it should be a small number to decrease the computational burdens. $N_{\max}$ denotes the upper bound of the iteration times. a_1 denotes a threshold to distinguish the degree of diverging. In this paper, χ^2 is set to 4.61 for a 90% confidence level of the two-order observation function, and a_1 will be discussed in the numerical test.

The pseudocode of the proposed resampling is shown in **Algorithm 1** where the operator $W()$ denotes the weight of the corresponding particle, and it is calculated as (32). The operator $BA(\cdot)$ denotes the particles after the proposed intelligent resampling step, and q denotes a random number drawn from the uniform distribution of U(0,1). The basic parameters of the resampling step are selected as suggested in [14], and they are shown in Table 1.

TABLE 1. Resampling parameters.

Parameter	Value	Quantity
A_i^0	0.9	Initial loudness for the *i*-th bat
α, γ	0.9	Constant numbers
r_i^0	0.1	Initial pulse rate for the *i*-th bat
ε	0.001	Random vector drawn from [-1,1]
$[f_{\min}, f_{\max}]$	[0, 1]	Frequency range
$[N_{\min}, N_{\max}]$	[3, 10]	Range of iteration times

Algorithm 1 Bat-Inspired Resampling

Inputs: p, $\{s_t^{[m]}, \hat{w}_t^{[m]}\}$, M

Results: $BA(\bar{s}_t)$

```
for ($itr$=0; $itr < p$;$itr$ ++)
    for $i = \alpha_{th}$ to $M$    //Resample on small-weight particles
        //Generate new solutions
        $f_i = f_{\min} + (f_{\max} - f_{\min}) \times U(0, 1)$
        $v_i' = v_i + (s_i - s_{best}) \times f_i$    //$v_i$ is set to 0 for initialization
        $s_i' = s_i + v_i'$ (Global search)
        //Random walk around a selected best solution
        if $q > r_i$
            $s_i' = s_{best} + \varepsilon A_{avg}$ (Local search)
        end if
        //Update the bat population
        if $W(s_i') > W(s_i)$ // Update the bat population
            $A_i = \alpha A_i$    //$A_i$ is firstly substituted by $A_i^0$
            $r_i = r_i^0(1 - e^{-\gamma \cdot itr})$
            $s_i = s_i'$    // Update the bat location
        end if
    end for
end for
normalize particles as (31)
update the optimized robot state as (32)
reset particle as $\hat{w}_t^{[m]} = 1/M$
```

E. OPTIMIZATION PROCESS OF PARTICLES IN IFASTSLAM

The schematic diagram of improving particle distribution in IFastSLAM is demonstrated in Fig.3, in which a larger size and darker color particle corresponds to a particle that has larger weight. Compared to the conventional methods by sampling from the prior distribution [27], [28], IFastSLAM samples from the proposal distribution generated by SRCKF, and thus the particles can tend to a higher posterior PDF region. Additionally, the intelligent resampling can optimize the particles with small-weight and drives the particles to better locations, i.e., the weights of the optimized particles

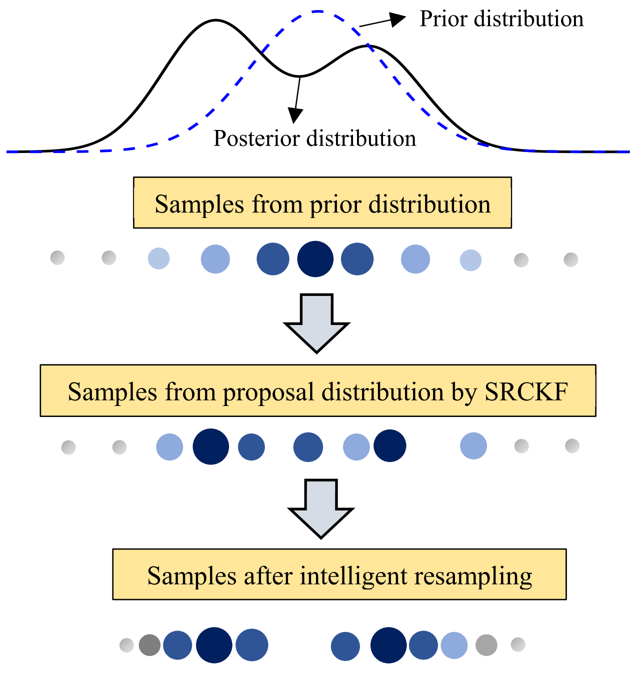

Prior distribution
Posterior distribution
Samples from prior distribution
Samples from proposal distribution by SRCKF
Samples after intelligent resampling

FIGURE 3. Schematic diagram of improving particle distribution in IFastSLAM.

TABLE 2. Importance sampling function and resampling method.

Algorithm	Importance sampling function	Resampling
SRUFastSLAM	Square root unscented Kalman filter	Systematic resampling
STSRCDFast-SLAM	Strong tracking square root central difference Kalman filter	Adaptive partial systematic resampling
IFastSLAM	Square root cubature Kalman filter	Adaptive bat-inspired resampling

become larger and the particle locations become closer to the high posterior PDF region than that before resampling. Therefore, improved by the proposed algorithm, IFastSLAM can theoretically better estimate the robot state. The verification of the algorithm is demonstrated in the following sections.

IV. NUMERICAL TESTS IN THE SIMULATOR

To evaluate the performance of the proposed IFastSLAM, it is compared to the square root unscented FastSLAM and strong tracking square root central difference FastSLAM in the simulator [17], [18], which is developed by Tim Bailey's group [19]. The composition of three algorithms is demonstrated in Table 2.

A. TEST ENVIRONMENT

In the simulator, the robot moves at a speed of $3m/s$ with a maximum steering angle of 30°. Moreover, the robot has a 4-m wheel base and a range-bearing sensor with a maximum range of 20-m and a 180° frontal view. The number of particles is set to 20. Fig.4 demonstrates the designed trajectory of the robot and landmarks location in

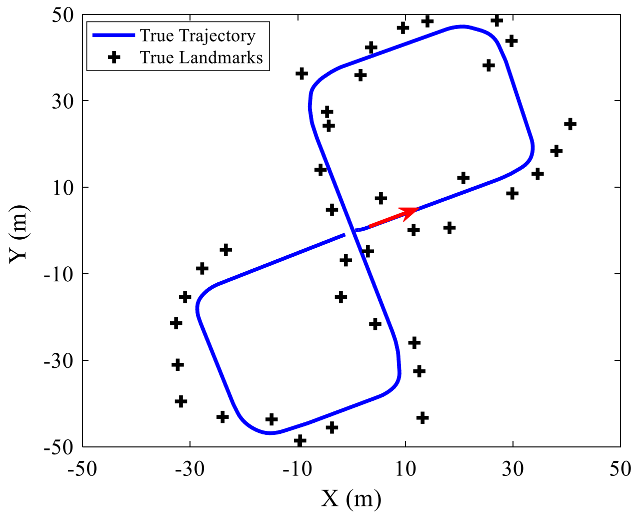


FIGURE 4. True trajectory and landmarks location (The direction of the red arrow denotes the motion direction of the robot).

TABLE 3. Noise levels.

Noise Levels	Control Noise	Observation Noise
Level A	(σ_v=0.3m/s, σ_g= 2.0°)	(σ_p=0.5m, σ_θ= 2.0°)
Level B	(σ_v=0.3m/s, σ_g= 2.0°)	(σ_p=0.8m, σ_θ= 3.0°)
Level C	(σ_v=0.8m/s, σ_g= 2.0°)	(σ_p=0.8m, σ_θ= 3.0°)
Level D	(σ_v=1.0m/s, σ_g= 3.0°)	(σ_p=0.8m, σ_θ= 3.0°)

the test environment. The cross-shaped points denote the known stationary landmarks in the simulation environment. The control and observation frequencies are set to 40Hz and 5Hz, respectively. For each test, the results are obtained over 20 Monte Carlo runs, and each run is carried out with 5 loops. The data association is assumed known throughout the process.

Here, four noise levels are used to evaluate the performance (stability and accuracy) of the proposed algorithm, and they are demonstrated in Table 3. Based on the given noise levels, the algorithm can be tested under different control noise with the same measurement noise, or under different control noise with the same measurement noise.

B. NUMERICAL TESTS

1) PARAMETER DESIGN

To determine an appropriate parameter of a_1 in (54), the performance of different a_1 are tested within the range of [2], [8] under four noise levels. Fig.5 demonstrates the root-mean-square error (RMSE) of different test conditions. It can be observed that when $a_1 = 2$ or $a_1 = 3$, the RMSE of the robot location can be reduced to a good result. Therefore, a_1 is set at 3 as it has possibly less iteration times.

2) PERFORMANCE OF INTELLIGENT RESAMPLING

To test the performance of proposed adaptive bat-inspired resampling (ABR), it is compared with the systematic

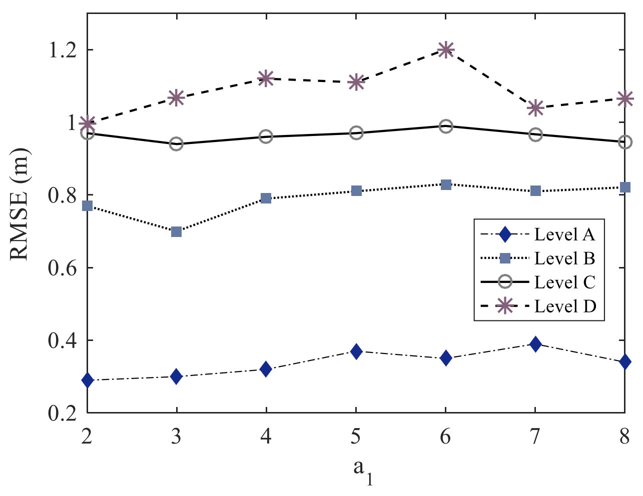


FIGURE 5. **RMSE of robot location using different a_1.**

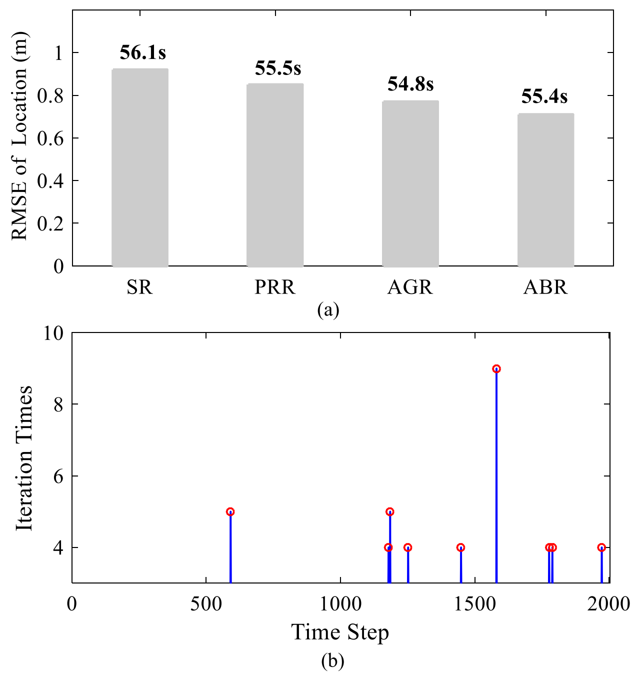


FIGURE 6. **(a) RMSE and cost time of each algorithm. (b) Iteration times of proposed resampling.**

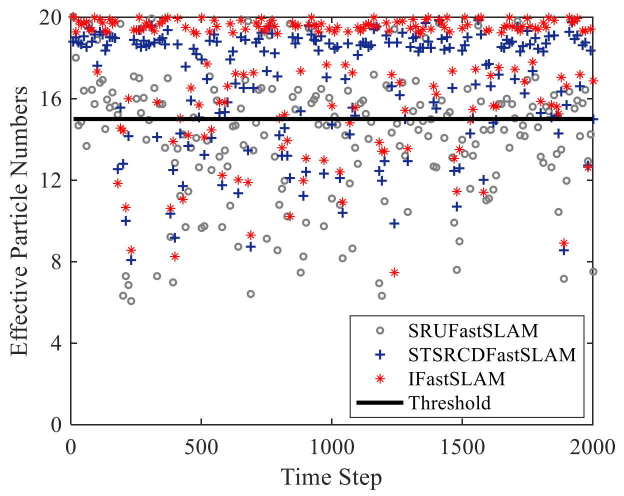


FIGURE 7. **Comparison of effective particles.**

resampling (SR) [13], partial rank-based resampling (PRR) [17], and adaptive genetic resampling (AGR) under noise level B [22]. Fig.6 (a) demonstrates the RMSE and cost time of IFastSLAM with different resampling methods. It can be observed that IFastSLAM with ABR has the best accuracy among four algorithms, and reduces the RMSE by 7.8%, 16.5% and 22.8% compared to that with AGR, PRR and SR, respectively. This is because the ABR has a better search ability compared to AGR as it combines the main advantages of genetic algorithms and particle swarm optimization. Additionally, the ABR can adaptively determine the particles that needed to be resampled rather than using an empirical threshold in PRR. In contrast to SR, the ABR does not contain redundant particles due to the bat optimization and thus can better address the particle impoverishment problem. It's observed that the IFastSLAM with ABR has the similar CPU running time compared to that with PRR, and reduces the computational time by 2.5% compared to that with SR. Although the algorithm with ABR has better accuracy than that with AGR, the iteration process adds additional running time to the algorithm. On the whole, the proposed intelligent resampling has better comprehensive performance compared to the other three resampling methods.

Fig.6 (b) illustrates the iteration times of the intelligent resampling during the simulation process. When the filter updates in a normal condition, the tuning strategy will not be triggered, and the iteration times maintain 3. When the filter diverges, the iteration times will be adjusted according to the degree of diverging. Note that the tuning strategy is rarely triggered under given conditions because the proposed resampling enhances the performance of FastSLAM and suppresses filter diverging.

In Fig.7, the effective particles of SRUFastSLAM, STSRCDFastSLAM and IFastSLAM are presented to evaluate the degeneracy degree of the filter. It can be observed that the effective particles of IFastSLAM are mostly higher than the other two algorithms at each time step. More specifically, the average effective particles of IFastSLAM is 24.7% and 8.3% larger than SRUFastSLAM and STSRCDFastSLAM respectively, i.e., IFastSLAM has the best anti-degeneracy performance among three algorithms.

3) LOCATION PERFORMANCE

Fig.8 shows the RMSE of the robot location under noise level B. It's observed that the IFastSLAM has the smallest location RMSE during the given time. This is mainly because the intelligent resampling maintains the diversity of particles and suppresses the filter diverging, improving the location accuracy of the robot.

Fig.9 demonstrates the estimated paths and feature location of SRUFastSLAM, STSRCDFastSLAM and

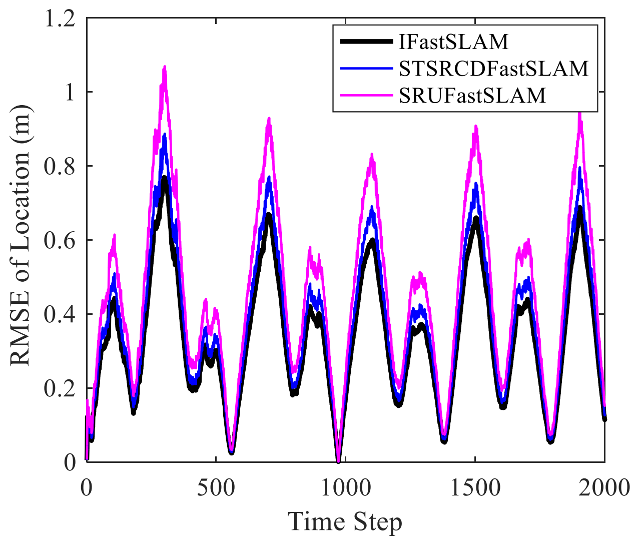


FIGURE 8. **RMSE of robot position.**

IFastSLAM, respectively. It is observed that the IFastSLAM has better robot and feature location accuracy compared to the other two algorithms. This result corresponds to the result in Fig.8.

Fig.10 demonstrates the RMSE of the estimated robot position under different particles and noise levels. It's observed that the RMSE and variance of the robot location decrease as the number of particles increases. Moreover, the IFastSLAM has a better accuracy than the SRUFastSLAM and STSRCD-FastSLAM algorithms with the same number of particles. It's noted that the RMSE of IFastSLAM with 5 particles is smaller compared to that of SRUFastSLAM and STSRCDFastSLAM with 30 particles. This result reveals that the less particles are needed for IFastSLAM to maintain a specified location accuracy. Based on the test results from the given conditions, it is concluded that the proposed IFastSLAM has better estimation accuracy than SRUFastSLAM and STSRCDFastSLAM.

4) COMPUTATIONAL COST

The computational cost of the proposed algorithm is analyzed using Matlab software on Intel(R) Core (TM) i7-7700 CPU@3.6GHz PC. The CPU running time of each algorithm is utilized to evaluate the computational complexity of the algorithms. As shown in Table 4, the CPU running time of three algorithms increases as the number of particles increases. Because the proposed IFastSLAM only operates the resampling step on small-weight particles, IFastSLAM can averagely improve the computational efficiency by 3.3% compared to SRUFastSLAM with systematic resampling. Although the proposed resampling step has an iteration process, it can improve the quality of all particles and thus reduce the possibility of executing the resampling step; Therefore, the computational efficiency of the IFast-SLAM algorithm is comparable to STSRCDFastSLAM even if the resampling step of STSRCDFastSLAM doesn't need iteration.

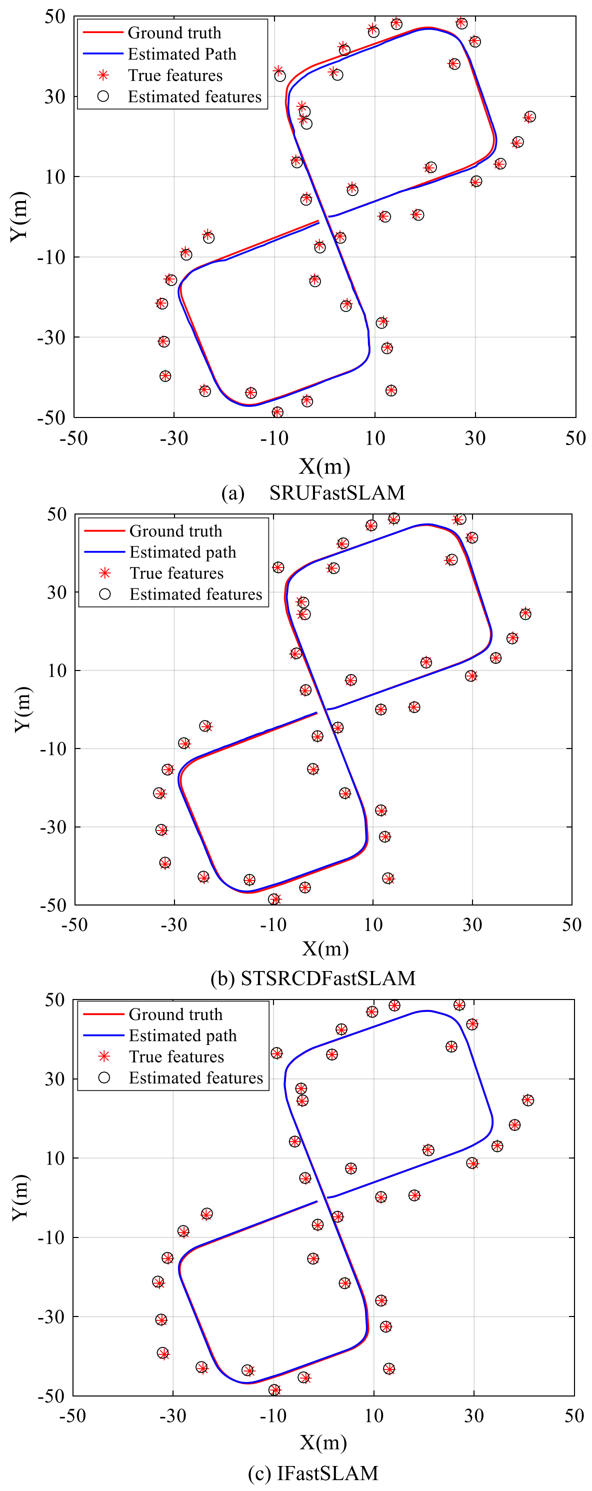


FIGURE 9. **Comparison of estimated paths of SRUFastSLAM and IFastSLAM.**

5) FILTER CONSISTENCY

Here, the normalized estimation error squared (NEES) is used to evaluate the consistency of the filter, and it is given by [17], [18]

$$\varepsilon_t = (s_t - \hat{s}_t)^T P_t^{-1} (s_t - \hat{s}_t) \tag{55}$$

TABLE 4. CPU running time/RMSE with different particles.

Particle Number	SRUFastSLAM		STSRCDFastSLAM		IFastSLAM	
	Time (s)	RMSE (m)	Time (s)	RMSE (m)	Time (s)	RMSE (m)
5	15.6	0.90	15.1	0.78	14.9	0.61
15	43.4	0.71	42.5	0.62	42.4	0.41
30	85.9	0.65	83.2	0.49	83.0	0.30

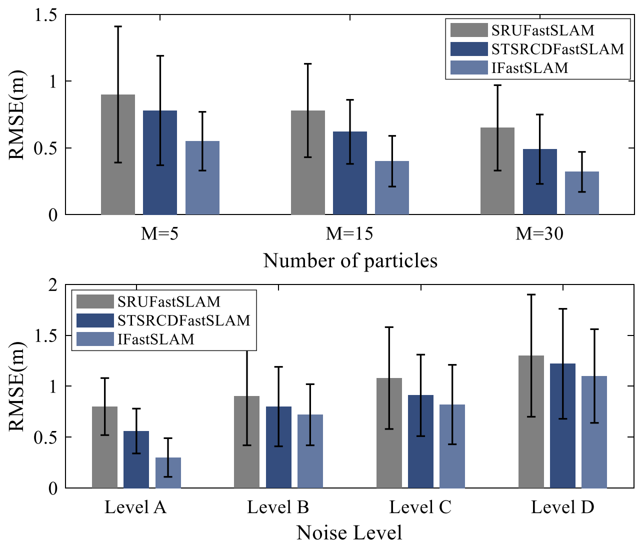


FIGURE 10. RMSE of the robot location under different noise levels and particles.

where s_t, $\hat{s}_t$ and P_t represent the truth, estimated, and the covariance of the robot trajectory, respectively. Filter consistency is calculated by Monte Carlo runs. With N_R runs, the average NEES (ANESS) is calculated as follows

$$\bar{\varepsilon}_t = \frac{1}{N_R}\sum_{i=1}^{N_R} \varepsilon_{it}. \tag{56}$$

For the 3-dimensional robot pose, the 95%-confidence level is bounded by interval [2.36, 3.72]. As shown in Fig.11, the filter consistency of the proposed method is better than SRUFastSLAM and STSRCDFastSLAM. More specifically, 94.0% ANEES of IFastSLAM locates in the bounded region, and it is larger than that of 38.5% of SRUFastSLAM and 24.6% of STSRCDFastSLAM.

V. VERIFICATION USING REAL-WORLD DATASET

In this section, the performance of IFastSLAM is compared to SRUFastSLAM and STSRCDFastSLAM using the datasets collected in Car Park and Victoria Park, respectively [19], [20]. The robot used for collecting datasets was equipped with a wheel encoder (ROD-430), GPS (Ashtech GG24), and a laser radar (SICK LMS 221) with a 180° frontal field-of-view. Here, the individual compatibility nearest neighbor (ICNN) test with a 2σ acceptance region is used for data association of features.

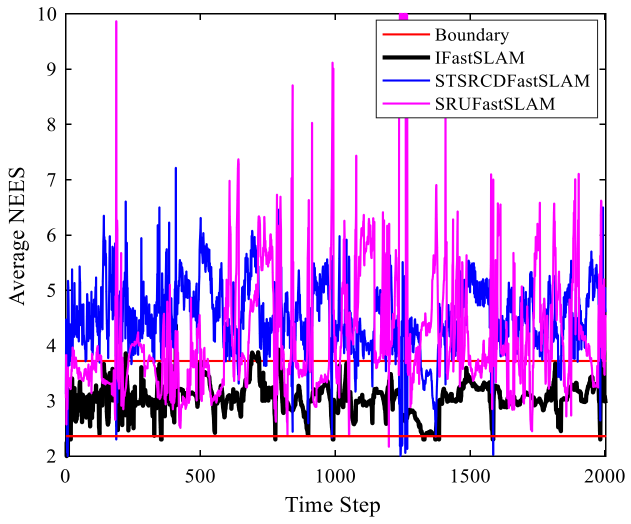


FIGURE 11. Comparison pf the average NEES.

TABLE 5. RMSE of robot position and CPU running time.

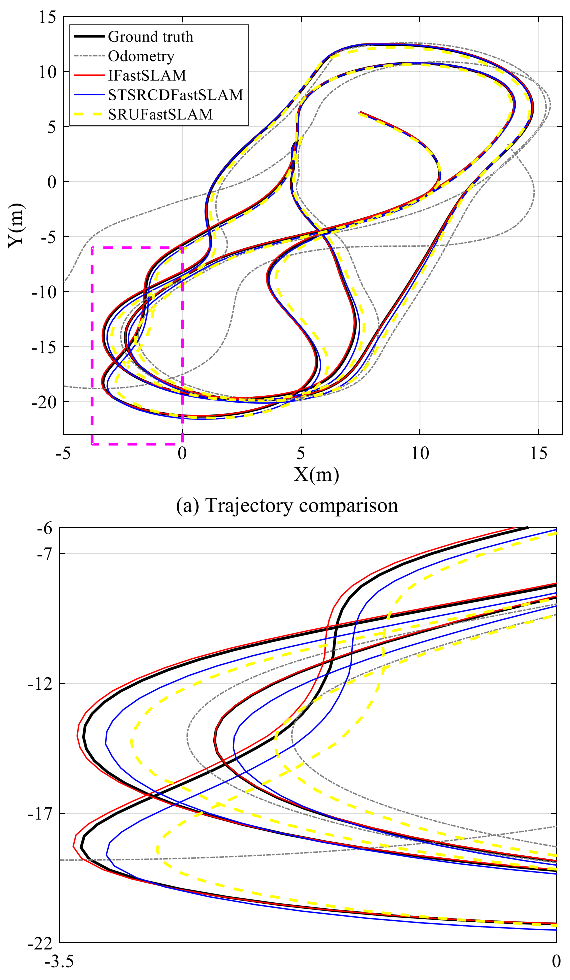
Dataset / Algorithm	In Car Park		In Victoria Park	
	RMSE (m)	Cost time (s)	RMSE (m)	Cost time (s)
SRUFastSLAM	0.56	108.8	11.1	1281.4
STSRCDFastSLAM	0.49	103.0	9.8	1236.2
IFastSLAM	0.42	102.4	8.8	1225.0

In the tests, the GPS provided the ground truth of the robot and stationary landmarks. The wheel encoders were utilized to measure the steering angle and the velocity of the robot. In Car Park, some man-made rods covered with reflective tape were used to be the environmental landmarks. In Victoria Park (Sydney, Austria), the robot moved around covering a path of 4 km approximately. Due to the obstruction by trees and buildings, the GPS data was sometimes unavailable. Nevertheless, the ground-truth position of the robot given by GPS was good enough to verify the estimated trajectory.

Fig.12 and Fig.13 (a) demonstrate the odometry and GPS paths of the robot in Car Park and Victoria Park, respectively. Without the laser information, the estimated path of the robot does not well match with the GPS path. It's observed that the estimated trajectories of IFastSLAM have better accuracy than that of the SRUFastSLAM and STSRCDFastSLAM in two benchmark environments as its estimated trajectories better coincide with the GPS path.

Table 5 shows the RMSE of the robot trajectory and CPU running time of three algorithms. The test result demonstrates that IFastSLAM improves the localization accuracy by 25.0% and 14.3% compared to SRUFastSLAM and STSRCDFast-SLAM when using Car Park dataset, and improves the localization accuracy by 20.7% and 10.2% when using Victoria Park dataset. These results correspond to the results shown in Fig.12 and Fig.13.

Additionally, compared to SRUFastSLAM, the proposed algorithm reduces the running time by 5.9% and 4.4% in Car Park and Victoria Park, respectively. Compared to STSRCDFastSLAM, IFastSLAM also has a competitive



(a) Trajectory comparison

(b) Enlarged view of the magenta bounded region.

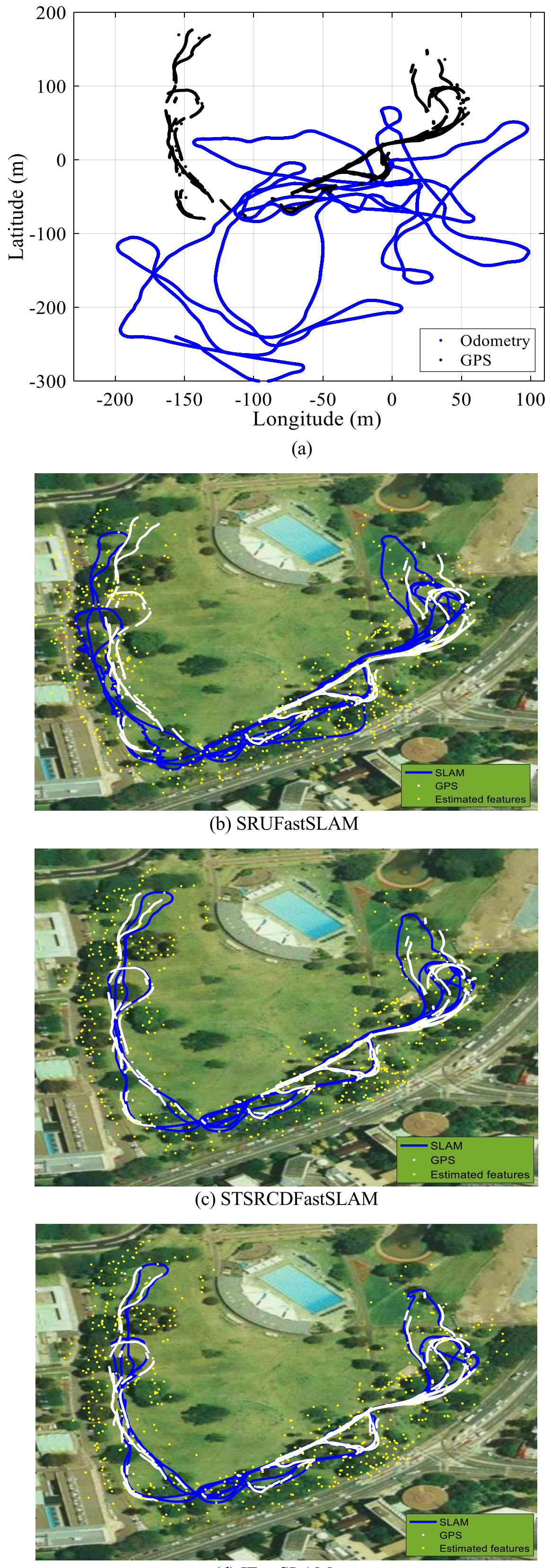
FIGURE 12. Trajectory comparison using car park dataset.

computational efficiency. These results correspond to the test results using the simulator.

VI. EXPERIMENT IN UNDERWATER ENVIRONMENT

To further validate the performance of IFastSLAM, it is tested in an underwater environment (swimming pool) using a four degree-of-freedom (DOF) underwater robot, as shown in Fig.14.

The underwater robot is equipped with a gyroscope (HWT901B), two cameras (SY002HD), four thrusters (ROVMAKER) and an ultra-wide bandwidth (UWB) positioning module (DW1000) whose localization error is within 10 cm. The buoyancy of the underwater robot is adjusted to let the UWB antenna just come out of the water so that the robot can receive localization information according to the position of three UWB stations beside the pool and approximately move in a horizontal plane. The UWB stations are shown in Fig.15. Here, the UWB positioning system is regarded as the ground truth reference system to evaluate the localization accuracy of the proposed algorithm.



(a)

(b) SRUFastSLAM

(c) STSRCDFastSLAM

(d) IFastSLAM

FIGURE 13. Trajectory comparison using victoria park dataset.

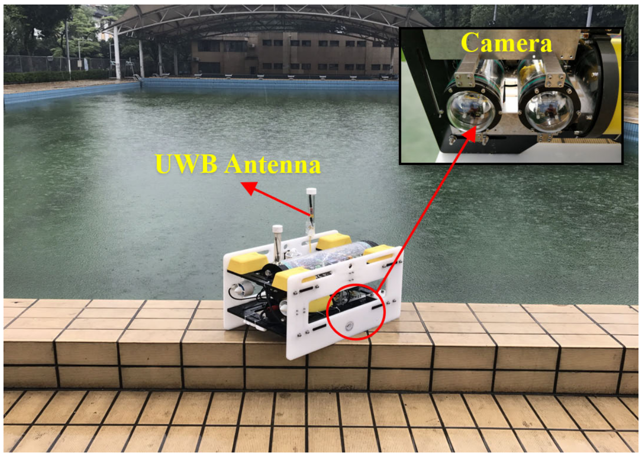


FIGURE 14. 4-DOF underwater robot.

FIGURE 15. UWB localization staions.

To apply the proposed algorithm to the underwater robot, the process state function and measurement function are changed to the following form

$$\chi_t^{[i][m]} = \underbrace{\begin{bmatrix} \chi_{x,t-1}^{[i][m]} \\ \chi_{y,t-1}^{[i][m]} \\ \chi_{\theta,t-1}^{[i][m]} \end{bmatrix}}_{\chi_{t-1}^{[i][m]}} + \begin{bmatrix} u_n \cos(\chi_{\theta,t-1}^{[i][m]})\Delta t + v_n \sin(\chi_{\theta,t-1}^{[i][m]})\Delta t \\ u_n \sin(\chi_{\theta,t-1}^{[i][m]})\Delta t - v_n \cos(\chi_{\theta,t-1}^{[i][m]})\Delta t \\ \omega_n \Delta t \end{bmatrix} \quad (57)$$

$$z_t = \begin{bmatrix} \chi_{x,t-1}^{[i][m]} \\ \chi_{y,t-1}^{[i][m]} \\ \chi_{\theta,t-1}^{[i][m]} \end{bmatrix} \quad (58)$$

where (u_n, v_n) and ω_n denote the linear and angular velocity of the underwater robot, respectively. The linear velocity is obtained via a model-based velocity prediction method of our previous work [29], and the angular velocity is provided by the gyroscope. Parameter z_t denotes the measurement vector at time step t, and it is given by the open-source binocular visual SLAM method [30].

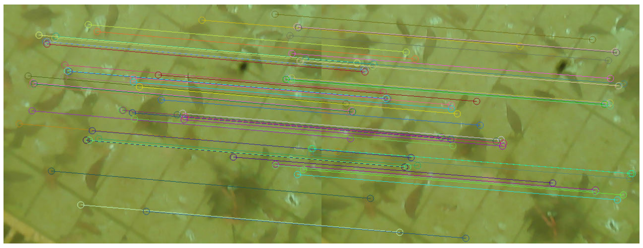


FIGURE 16. Underwater matched features.

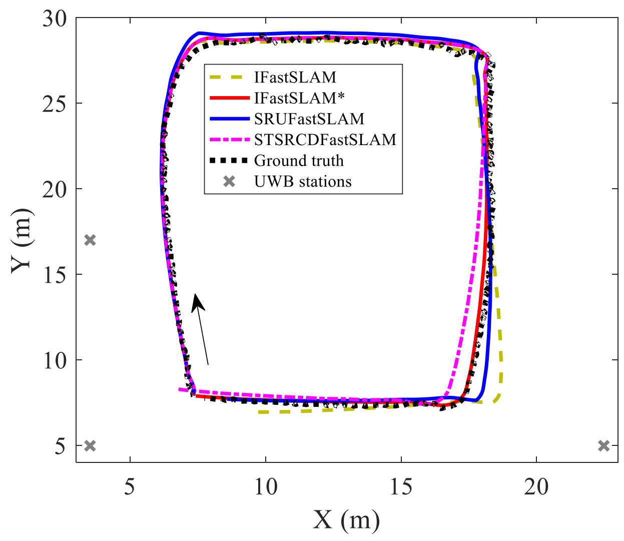


FIGURE 17. Comparison of localization performance in the pool experiment.

Fig.16 demonstrates the feature matching result of two images synchronously captured from the cameras. The major features in the observed environment are leaves and mud on the water bottom. According to the SLAM method, the pose and location of the underwater robot can be calculated, and they are taken as the actual measurement in the algorithm. One should note that the captured images should be calibrated before feature extraction in order to reduce image distortion caused by the reflection from the media of water and watertight shell, and this process is achieved using Zhang's calibration method [31]. Moreover, the maximum extracted features are set to 100 to maintain a good computational efficiency of the algorithm. The update rate of the algorithm is 6Hz in the experiment.

In real-world missions, mobile robots would inevitably acquire some unreliable measurements. For example, the underwater robot would capture blurry and degraded images in high-turbidity and low-illumination environment; as a result, different features may be identified as the same one, causing feature mismatch and inaccurate robot localization. To improve the robustness of the algorithm, an additional examination step is added to the algorithm based on the Mahalanobis distance. The Mahalanobis distance in (51)-(53) is used to determine the iteration times of the bat-inspired resampling. Here, the Mahalanobis distance is repeatedly calculated after the resampling step. If $D_t/\chi^2 \leq 1$, the measurement is deemed to be reliable; otherwise, the measurement is deemed to be unreliable, and the algorithm only executes the state prediction process of (11)-(18). The algorithm with the outlier removal process is denoted as IFastSLAM*.

Fig.17 demonstrates the trajectories of the proposed algorithm and comparison results with SRUFastSLAM and STSRCDFastSLAM. It's observed that the IFastSLAM algorithm with the outlier removal process has the best localization accuracy. Moreover, the proposed algorithm reduces the computational time by 3.8% and 1.4% compared to SRUFastSLAM and STSRCDFastSLAM respectively in the experiment, further verifying the advantages of the proposed algorithm.

VII. CONCLUSION

This paper proposes an intelligent filter-based SLAM to improve localization performance of mobile robots, and it is verified by comparing to two state-of-the-art FastSLAM algorithms. The proposed algorithm can not only be applied to terrestrial scenarios, but also aerial and underwater cases, as long as the process function and measurement function can be established appropriately.

One should note that IFastSLAM is more suitable for the environment with sparse features, such as trees, beacons, and seafloor rocks. This is because a particle filter-based algorithm cannot well address a mass of features by updating each particle in real-time. Furthermore, to maintain robustness and efficiency of the algorithm, it is necessary to remove unreliable/outlier measurements and limit the maximum number of measurements.

The main contributions of this paper are concluded as follows:

1) The SRCKF is fused to the FastSLAM algorithm for stepwise estimation of robot and feature locations.
2) An adaptive bat-inspired resampling is proposed to overcome the problems of particle degeneracy and impoverishment, and it proves to have better optimization effects compared to the SR, PRR and AGR methods.
3) The proposed IFastSLAM algorithm can remove outlier measurements and has better localization accuracy and computational efficiency compared to SRUFastSLAM and STSRCDFastSLAM in the simulation, real-world dataset test and pool experiment.

The future study is going to focus on optimizing the framework of the algorithm for real-time applications even if the robot is in a feature-rich environment.

REFERENCES

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

[2] A. Mallios, P. Ridao, D. Ribas, F. Maurelli, and Y. Petillot, "EKF-SLAM for AUV navigation under probabilistic sonar scan-matching," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, Oct. 2010, pp. 4404–4411.

[3] 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.

[4] A. G. Thallas, E. G. Tsardoulias, and L. Petrou, "Topological based scan matching—Odometry posterior sampling in RBPF under kinematic model failures," *J. Intell. Robot. Syst.*, vol. 91, no. 3, pp. 543–568, 2017.

[5] K. Y. K. Leung, F. Inostroza, and M. Adams, "Multifeature-based importance weighting for the PHD SLAM filter," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 52, no. 6, pp. 2697–2714, Dec. 2016.

[6] A. Thallas, E. Tsardoulias, and L. Petrou, "Particle filter—Scan matching hybrid SLAM employing topological information," in *Proc. 24th Medit. Conf. Control Automat. (MED)*, Jun. 2016, pp. 226–231.

[7] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM: A factored solution to the simultaneous localization and mapping problem," in *Proc. 18th National Conf. Artif. Intell.*, Jul. 2002, pp. 593–598.

[8] J. Luo and S. Qin, "A fast algorithm of simultaneous localization and mapping for mobile robot based on ball particle filter," *IEEE Access*, vol. 6, no. 6, pp. 20412–20429, Mar. 2018.

[9] Q.-L. Li, Y. Song, and Z.-G. Hou, "Neural network based FastSLAM for autonomous robots in unknown environments," *Neurocomputing*, vol. 165, pp. 99–110, Oct. 2015.

[10] R. Havangi, "A mutated FastSLAM using soft computing," *Ind. Robot Int. J.*, vol. 44, no. 4, pp. 416–427, Jun. 2017.

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

[12] I. Arasaratnam and S. Haykin, "Cubature Kalman filters," *IEEE Trans. Autom. Control*, vol. 54, no. 6, pp. 1254–1269, Jun. 2009.

[13] L. Tiancheng, B. Miodrag, and D. M. Petar, "Resampling methods for particle filtering: Classification, implementation, and strategies," *Signal Process. Mag.*, vol. 32, no. 3, pp. 70–86, May 2015.

[14] M. A. Al-Betar, M. A. Awadallah, H. Faris, X. S. Yang, and A. T. Khader, "Bat-inspired algorithms with natural selection mechanisms for global optimization," *Neurocomputing*, vol. 273, pp. 448–495, Jan. 2018.

[15] S. Jafarzadeh, C. Lascu, and M. S. Fadali, "Square root unscented Kalman filters for state estimation of induction motor drives," *IEEE Trans. Ind. Appl.*, vol. 49, no. 1, pp. 92–99, Feb. 2013.

[16] B. Zheng and Z. Zhang, "An improved EKF-SLAM for Mars surface exploration," *Int. J. Aerosp. Eng.*, vol. 2019, pp. 1–9, Apr. 2019.

[17] R. Havangi, H. D. Taghirad, M. A. Nekoui, and M. Teshnehlab, "A square root unscented FastSLAM with improved proposal distribution and resampling," *IEEE Trans. Ind. Electron.*, vol. 61, no. 5, pp. 2334–2345, May 2014.

[18] 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 T. Intell. Transp.*, vol. 17, no. 11, pp. 3110–3120, Nov. 2016.

[19] *Homepage of Tim Bailey*. [Online]. Available: http://www-personal.acfr.usyd.edu.au/tbailey/

[20] *ACFR Dataset*. [Online]. Available: http://www-personal.acfr.usyd.edu.au/nebot/dataset.htm

[21] M. Montemerlo and S. Thrun, *FastSLAM: A Scalable Method for the Simultaneous Localization and Mapping Problem in Robotics*. Berlin, Germany: Springer-Verlag, 2007.

[22] M. Lin, C. Yang, and D. Li, "An improved transformed unscented FastSLAM with adaptive genetic resampling," *IEEE T. Ind. Electron.*, vol. 66, no. 5, pp. 3583–3594, May 2019.

[23] S. A. Holmes, G. Klein, and D. W. Murray, "An O(N^2) square root unscented Kalman filter for visual simultaneous localization and mapping," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 7, pp. 1251–1263, Jul. 2009.

[24] C. Kim, R. Sakthivel, and W. K. Chung, "Unscented FastSLAM: A robust and efficient solution to the SLAM problem," *IEEE Trans. Robot.*, vol. 24, no. 4, pp. 808–820, Aug. 2008.

[25] S. Yin and X. Zhu, "Intelligent particle filter and its application to fault detection of nonlinear system," *IEEE Trans. Ind. Electron.*, vol. 62, no. 6, pp. 3852–3861, Jun. 2015.

[26] J. M. Pak, C. K. Ahn, Y. S. Shmaliy, and M. T. Lim, "Improving reliability of particle filter-based localization in wireless sensor networks via hybrid particle/FIR filtering," *IEEE Trans. Ind. Inform.*, vol. 11, no. 5, pp. 1089–1098, Oct. 2015.

[27] S. Yin, X. Zhu, J. Qiu, and H. Gao, "State estimation in nonlinear system using sequential evolutionary filter," *IEEE Trans. Ind. Electron.*, vol. 63, no. 6, pp. 3786–3794, Jun. 2016.

[28] A. Khorshidi and A. M. Shahri, "GA-inspired particle filter for mitigating severe sample impoverishment," in *Proc. 4th Int. Conf. Control, Instrum., Automat. (ICCIA)*, Jan. 2016, pp. 377–382.

[29] M. Lin, C. Yang, and D. Li, "An improved model-based velocity prediction method of underwater vehicles using a hybrid estimation strategy," *IEEE Trans. Ind. Inform.*, 2019.

[30] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An open-source slam system for monocular, stereo, and RGB-D cameras," *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1255–1262, Oct. 2017.

[31] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 11, pp. 1330–1334, Nov. 2000. [Online]. Available: http://research.microsoft.com/en-us/um/people/zhang/Papers/TR98-71.pdf, accessed Oct. 30, 2014.

MINGWEI LIN (M'18) received the B.S. degree in mechanical engineering and automation from Fuzhou University, China, in 2014, and the Ph.D. degree in mechatronic engineering from Zhejiang University, China, in 2019, where he is currently a Postdoctoral Research Fellow with the State Key Laboratory of Fluid Power and Mechatronic Systems.

His research interests include simultaneous localization and mapping (SLAM), sensor fusion, AUV docking technology, and their applications to deep-sea exploration.

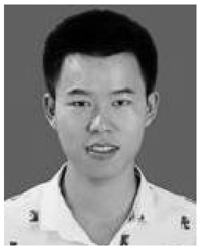
CANJUN YANG received the B.S. and M.S. degrees in mechanical engineering from the Nanjing University of Aeronautics and Astronautics, China, in 1991 and 1994, respectively, and the Ph.D. degree in mechanical engineering from Zhejiang University, China, in 1997, where he has been a Faculty Member with the State Key Laboratory of Fluid Power and Mechatronic Systems, since 1997.

In 2004, he visited Minnesota University as an Academic Visitor. Since 2004, he has been a Professor with the Institute of Mechatronic and Control Engineering, Zhejiang University. His research interests include navigation, man–machine intelligent systems, mechatronic devices, and their applications to deep-sea exploitation and exploration.

Dr. Yang received the National Technology Invention Award (Second Prize), in 2009, the University Technology Invention Award (First Prize) of the Ministry of Education of China, in 2006, the Science and Technology Award of Zhejiang Province (First Prize), in 2006, and the Zhejiang Youth Science and Technology Award, in 2009. He was selected to the Program for New Century Excellent Talents (NCET) in University and the New Century 151 Talent Project of Zhejiang Province (First Class). He is currently a member of the National Committee Submersible of Standardization and the trustee of China Innovation Strategic Alliance of Rehabilitation Technical Aids.

DEJUN LI received the B.S. and M.S. degrees in mechanical engineering from Tsinghua University, Beijing, China, in 1993 and 1996, respectively, and the Ph.D. degree in mechanical engineering from Hiroshima University, Japan, in 2003.

He was a Postdoctoral Research Fellow with the Graz University of Technology, Graz, Austria, from 2004 to 2006. He is currently a Professor with the Institute of Mechatronic and Control Engineering, Zhejiang University, Hangzhou, Zhejiang, China. His research interests include cable ocean observation, autonomous underwater vehicle (AUV), underwater docking, contactless power transmission, and its applications to deep-sea exploration.



GENGLI ZHOU received the B.S. degree in mechanical engineering from Xi'an Jiaotong University, China, in 2017. He is currently pursuing the M.S. degree in mechatronic engineering with the State Key Laboratory of Fluid Power and Mechatronic Systems, Zhejiang University, Hangzhou, China.

His research interests include the simultaneous localization and mapping (SLAM) and its applications to deep-sea exploration.