An Improved Transformed Unscented FastSLAM With Adaptive Genetic Resampling

Mingwei Lin, *Student Member, IEEE*, Canjun Yang, and Dejun Li

***Abstract*—Fast simultaneous localization and mapping (FastSLAM) is a well-known study for robot navigation. To enhance the performance of FastSLAM, an improved importance sampling is proposed in this paper based on the transformed unscented Kalman filter. The improvement is mainly composed of a novel fuzzy noise estimator, which can adjust the state and observation noises online according to the residual and related covariance, and thus mitigating the defects caused by model inaccuracy. In general, the FastSLAM algorithm suffers from the impoverishment problem since it is essentially a particle filter. Inspired by genetic optimization, an adaptive genetic resampling is proposed to substitute the conventional resampling step to overcome these defects. The proposed method, referred to as the improved transformed unscented FastSLAM, is compared with the unscented FastSLAM and the transformed unscented FastSLAM. The superiorities of the proposed method are verified by simulation and experiment under benchmark environments.**

***Index Terms*—Adaptive genetic algorithm (GA) resampling, FastSLAM, fuzzy noise estimator, particle filter (PF), simultaneous localization and mapping (SLAM), transformed unscented Kalman filter (TUKF).**

I. Introduction

SIMUTANEOUS localization and mapping (SLAM) is a well-known solution for robot navigation in unknown environments. SLAM addresses the problem of building a map from consecutive environmental features obtained from a moving robot. Meanwhile, the robot locates itself according to the map information.

Some basic frameworks, such as the extended Kalman filter SLAM (EKF-SLAM), unscented Kalman filter SLAM (UKF-SLAM), and FastSLAM, have been widely used for the SLAM applications [1]. Common EKF-SLAM suffers from the computationally expensive and filter inconsistency problems because of the linearization of nonlinear models. The UKF-SLAM employs the unscented transformation to approximate the state distribution instead of the linearization operation used in the EKF, and thus improving the filter accuracy. However, the most popular algorithm among these algorithms is the FastSLAM, an idea from the Rao–Blackwellized particle filter [2], [3]. This is because it has the reduced computational complexity acquired by factorizing the full posterior distribution into a product of landmark distributions and a robot path distribution only with a complexity of logarithmic scaling regarding the number of map features. To improve the performance of FastSLAM, many researchers focus on generating a better proposal distribution, such as employing the UKF [4], the central difference Kalman filter (CDKF) [5] and the cubature Kalman filter (CKF) [6] to be the importance sampling function. However, some drawbacks contained in these algorithms are inevitable. In the UKF, there are three scalar scaling parameters that should be well-adjusted. However, for the CDKF, it has the similar accuracy compared with the UKF but with only one adjustive parameter [5]. It has been approved that the CKF has better stability and accuracy than the UKF since all the cubature points have the equal constant positive weights without the central *sigma points* [7]. However, the CKF has inherent nonlocal sampling problems when the high dimension state is involved [8]. To address this problem, a transformed unscented Kalman filter (TUKF) is proposed in [8]. The weights of its sampling points are the same as those of cubature points and the radius of the sphere that bounds the new points does not increase with the state dimension. Moreover, the TUKF has better accuracy while the computational efforts are almost the same as CKF [9]; therefore, the TUKF is selected as the basis of the importance sampling function in this paper.

Despite the above-mentioned improvements, there exist some potential problems for the importance of sampling function. The typical one is the difficulty to acquire the accurate noise statistics of the state and measurement models in real-world applications; consequently, two models cannot perfectly match with each other. This inaccuracy leads to the improper measurement effect of the filter while modifying the predicted results (measurement update); therefore, the filter failure may occur after a certain running time [10] and the accuracy of the filter cannot be satisfactory. To solve this problem, some algorithms, such as the strong tracking filter (STF) [11] and the fuzzy inference system [12] are proposed to adjust the measurement noise. However, these two methods only consider the measurement noise effect on the residual and ignore the influence of process noise. In this paper, a fuzzy noise estimator (FNE) is proposed in the importance sampling function to adjust the measurement and process noises simultaneously. The proposed FNE works by referring

Manuscript received January 24, 2018; revised April 14, 2018, May 16, 2018, and June 6, 2018; accepted June 21, 2018. Date of publication July 16, 2018; date of current version December 28, 2018. This work was supported in part by the National Natural Science Foundation of China under Grant 41676089 and in part by the National High Technology Research and Development Program of China under Grant 2013AA09A414 and Grant 2014AA09A513. *(Corresponding author: Canjun Yang.)*

The authors are with the State Key Laboratory of Fluid Power and Mechatronic Systems, Zhejiang University, Hangzhou 310027, China (e-mail: mwlin_1023@163.com; ycj@zju.edu.cn; li_dejun@zju.edu.cn).

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

Digital Object Identifier 10.1109/TIE.2018.2854557

to the membership functions regarding the residual and related covariance. Given the proposed fuzzy rules, the FNE has better estimation accuracy of noise than that of STF [11] and fuzzy inference system (FIS) [12]. Moreover, the FNE has reduced computational burdens compared with the method of maximum posterior and random weighting (MPAW) introduced in [13].

Another way to improve the performance of FastSLAM is to optimize the resampling steps. Conventional resampling algorithms, including the systematic, stratified, and the residual resampling [14], [15], suffer from the impoverishment problems since the large-weight particles are repeatedly selected and the small-weight particles are deleted after resampling. As a result, the redundant particles cannot well-approximate the true state. Particle swarm optimization [16], [17] and a genetic algorithm (GA) [18], [19] are two commonly-used methods to maintain the diversity of particles before the resampling step. However, these intelligent methods generally have two disadvantages. The first one is that the optimization should be executed every period even if the algorithm currently has adequate effective particles without optimization. The second one is the undetermined scalar parameters, which should be tuned empirically. To solve these problems, an adaptive genetic resampling is proposed in this paper. This method utilizes a GA to substitute conventional resampling step; therefore, it is carried out only when the effective particles are less than the threshold. Furthermore, the crossover and mutation coefficients are provided according to the distribution of particle weights, in other word, they do not need any manual tuning.

To further enhance the algorithm, a popular square root filter with QR decomposition is used in the importance sampling function [17], [20]. The fused method for the importance sampling is referred to as the improved TUKF (ITUKF). The whole algorithm of SLAM is named as the ITUFastSLAM.

The rest of this paper is organized as follows. Section II introduces the background of FastSLAM and the *sigma points* of TUKF. Section III demonstrates the proposed fuzzy noise estimator. Section IV provides the framework of the ITUFastSLAM with the adaptive genetic resampling. In Section V, the simulation and experiment are conducted using the simulator [21] and datasets collected in Car Park and Victoria Park [22], respectively. Section VI provides some discussion and draws a conclusion.

II. Background

A. FastSLAM Problem

From a probabilistic view, SLAM estimates the posterior probability over the robot path along with the map [2]

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

where the path of the robot is given by $s^t = \{s_1, \ldots, s_t\}$, and θ denotes the map. Each landmark is denoted by θ_k for $k = 1, \ldots, N$ where N is the number of stationary landmarks possessed by the robot. $z^t = \{z_1, \ldots, z_t\}$ and $u^t = \{u_1, \ldots, u_t\}$ are the measurements and controls (or odometry information) up to time t, respectively. $n^t = \{n_1, \ldots, n_t\}$ is the data association result in which n_t determines the identity of the landmark observed at time t. The factorization idea used in the FastSLAM is demonstrated as follows [2]:

$$p(s^t, \theta \mid z^t, u^t, n^t) = p(s^t \mid z^t, u^t, n^t) \prod_{k=1}^{K} p(\theta_k | s^t, z^t, u^t, n^t). \tag{2}$$

Each particle m is of the form

$$S_t{}^{[m]} = \left\{ s^{t,[m]}, \mu_{1,t}^{[m]}, \sum_{1,t}^{[m]}, \ldots, \mu_{K,t}^{[m]}, \sum_{K,t}^{[m]} \right\} \tag{3}$$

where $[m]$ is the index of the particle ($m = 1, 2, \ldots M$), $s^{t,[m]}$ is the mth particle's path estimate, and $\mu_{K,\,t}^{[m]}$ and $\sum_{K,\,t}^{[m]}$ are, respectively, the mean and the covariance of the Gaussian distribution representing the Kth feature location of the mth particle. According to the FastSLAM 2.0, the robot pose s_t is sampled by [23]

$$s_t \sim p\left(s_t \mid s^{t-1,[m]}, z^t, u^t, n^t \right). \tag{4}$$

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

$$w_t{}^{[m]} = \frac{\text{target distribution}}{\text{proposal distribution}} = \frac{p\left(s^{t,[m]} \middle| z^t, u^t, n^t \right)}{p\left(s^{t-1,[m]} \middle| z^{t-1}, u^{t-1}, n^{t-1} \right) p\left(s_t^{[m]} \middle| s^{t-1,[m]} z^t, u^t, n^t \right)}. \tag{5}$$

The detailed derivation of the importance weights is shown in [2] and [23]. Finally, a resampling step is executed according to the computed weights.

B. Sigma Points of TUKF

The TUKF, which can address the nonlocal sampling problem inherently while maintaining the virtue of numerical stability for high dimension problems [8], is selected as the framework of proposed SLAM algorithm. The TUKF has a better performance compared with the UKF because it uses a set of *sigma points* with equal constant positive weights. The points set used for calculating *sigma points* are expressed as

$$\gamma = (\gamma_1, \gamma_2, \ldots, \gamma_{2L})_{L \times 2L} \tag{6}$$

where $\gamma_k = (\gamma_{k,1}, \gamma_{k,2}, \ldots, \gamma_{k,L})^T$, L denotes the dimension of the state, and $k = 1, 2, \ldots, 2L$. The element of γ_k is given by

$$\gamma_{k,2r-1} = \sqrt{2} \cos\left(\frac{(2r-1)k\pi}{L} \right), \quad \gamma_{k,2r} = \sqrt{2} \sin\left(\frac{(2r-1)k\pi}{L} \right) \tag{7}$$

where $r = 1, 2, \ldots, [L/2]$. If L is odd, $\gamma_{k,L} = (-1)^k$. $[L/2]$ is the greatest integer not exceeding $L/2$. The L-dimensional random variable x with mean $\hat{x}$ and covariance $\mathbf{P}_x$ is approximated by

$$\zeta_k = \hat{x} + \sqrt{\mathbf{P}_x}\gamma_k \tag{8}$$

in which ζ_k is the *sigma points* set.

III. Fuzzy Noise Estimator

A. Design of Fuzzy Noise Estimator

In this part, an FNE is introduced to mitigate the imperfect match between the process and measurement models. The main function of the proposed FNE is to adjust the noise according to the residual (sometimes referred to as innovation) and related covariance of observation. This idea is acquired according to the fact that the residual should be zero-mean Gaussian white noise if the filter is operated under perfect models. If the residual deviates from zero and locates in a small probability region of probability density function, the noise statistics are possibly inaccurate. However, it is difficult to formulize the mathematic relation between this inaccuracy and its possibility. Therefore, a fuzzy strategy is proposed to judge whether the noise statistics are accurate.

1) If the residual is nearly zero, the noise statistics are supposed to be accurate.
2) If the residual is greater than zero and the corresponding covariance is small, the noise statistics are deemed inaccurate and the noise should be decreased.
3) If the residual is smaller than zero and the corresponding covariance is small, the noise statistics are deemed inaccurate and the noise should be increased.
4) If the conditions are not satisfied with the above-mentioned three points, the accuracy of noise statistics are considered difficult to judge and the noise covariance remains unchanged.

The residual is defined as the difference between the actual observation z_t and the predicted observation $\hat{z}_t$ at time step t, and it is given by

$$r_t = z_t - \hat{z}_t. \tag{9}$$

To improve the reliability, the residual and corresponding covariance are averaged within a time window, respectively, and they are given by

$$\bar{r}_t = \frac{1}{q}\sum_{i=1}^{q} r_{t+i-q}, \bar{\mathbf{P}}_{r,t} = \frac{1}{q}\sum_{i=1}^{q} r_{t+i-q} r_{t+i-q}^T \tag{10}$$

where q is the length of the time window. The tuning method of the noises $\mathbf{Q}$ and $\mathbf{R}$ is formulized as

$$\mathbf{R}^* = (1+\lambda)\mathbf{R}, \quad \mathbf{Q}^* = (1+\lambda)\mathbf{Q} \tag{11}$$

where λ is calculated according to the aforementioned fuzzy strategy. Before defuzzification, the convergence of the importance sampling function (ITUKF) should be proved with the modified noise covariance, and it is provided in Section III-B. The proof reveals that the increment of noise covariance should be positive definite, in other word, $\lambda > 0$. Therefore, to satisfy the stability of the filter, in the second point of the aforementioned fuzzy strategy, the noise covariance should remain unchanged. The fuzzy membership functions are illustrated as Fig. 1 where the operator $\mathrm{trace}\,(\cdot)$ denotes the trace of the matrix.

The magnitude of denotation $\bar{r}_t^*$ is equal to $|\bar{r}_t|$, and the sign of $\bar{r}_t^*$ is determined by the sign of elements of residual. If all the elements are larger (smaller) than 0, the sign of $\bar{r}_t^*$ is positive (negative). To maintain appropriate robustness and sensitivity of fuzzy rule, the maximum value of fuzzy intersection is generally set to 0.5 as shown in Fig. 1 [10] (0.4–0.7 is acceptable according to the special demand). For each membership function, three fuzzy sets are used. According to the outputs of the membership functions and the aforementioned fuzzy strategy, a fuzzy rule table (see Table I) is proposed to calculate λ.

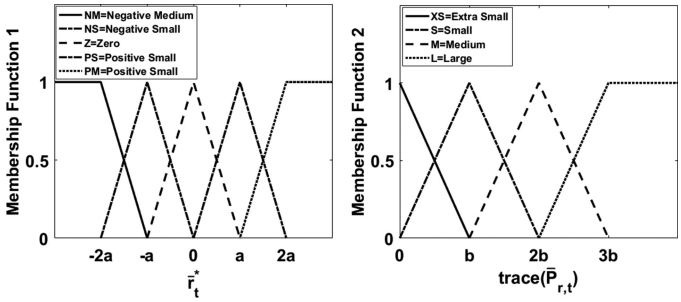

Fig. 1. Membership functions.

TABLE I
Fuzzy Rule Table for Calculating λ

$\bar{r}_t^*$ \ $\bar{\mathbf{P}}_{r,t}$	XS	S	M	L
NM	L	L	S	XS
NS	M	M	S	XS
Z	Z	Z	Z	Z
PS	Z	Z	Z	Z
PM	Z	Z	Z	Z

(*Note:* Z: Zero (0); XS: Extra Small (0.5c); S: Small (c); M: Medium (2c); L: Large (3c). The Corresponding λ is Given in Brackets Using Letter c).

The scale parameters a, b, and c determine the performance of proposed fuzzy estimator; therefore, the tuning range of them should be roughly specified before trial-and-error tests. First of all, the tuning range of scale parameters must satisfy the conditions of filter stability; therefore, the lower bound of c is 0. Moreover, the maximum and minimum values of a and b is obtained over a certain number of repeated tests without FNE, in other word, the maximum and minimum values of a and b during the tests are selected as the upper and lower bounds of their tuning range. As for the parameter c, its upper bound is obtained under the tests with the FNE using two groups: first, maximum a and b; second, minimum a and b. In other word, the maximum value of c during the test is selected as the upper bound of its tuning range. According to the obtained tuning range, different combinations of three scale parameters are tested as shown in Table II, which is acquired under 30 Monte Carlo runs with the initial noise of ($\sigma_\rho = 0.8\,\mathrm{m}$, $\sigma_\theta = 2.0^\circ$, $\sigma_v = 0.3\,\mathrm{m/s}$, $\sigma_g = 3.0^\circ$) and 10 particles. Fig. 2 demonstrates part of test groups. According to the test result, the group ($a = 0.4$, $b = 1$, $c = 0.05$) has the best accuracy and is used in this paper. The tuning procedure is a trial-and-error method, which should be adjusted when used for different systems. In other word, a different process and measurement models, or different

TABLE II
RMSE WITH DIFFERENT SCALE PARAMETERS (UNIT: METER)

Group		*c*		
a	***b***	0.02	0.05	0.08
0.2	0.5	0.24	0.19	0.31
0.2	1.0	0.25	0.21	0.35
0.4	0.5	0.27	0.23	0.33
0.4	1.0	0.25	0.17	0.33
0.6	0.5	0.26	0.24	0.40
0.6	1.0	0.25	0.19	0.39

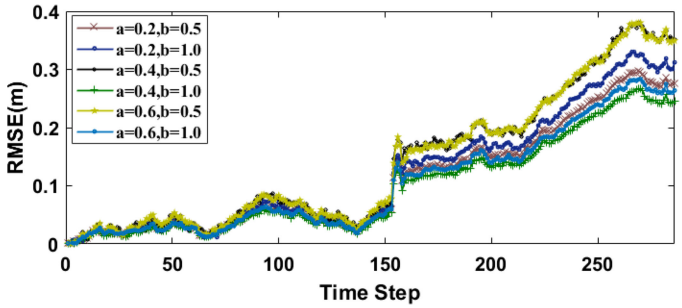

Fig. 2 Part of RMSE with different scale parameters. ($c = 0.05$).

importance sampling function may correspond to a different group of "optimal" scale parameters.

B. Stability Analysis of ITUKF

In this part, the stability of the proposed ITUKF, which contains a square root filter and a fuzzy noise estimator, is analyzed based on the bounded convergence of the conventional UKF [13], [24]. The square root filter only propagates the covariance of states by Cholesky decomposition, which is used to avoid truncation error caused by the processing unit and does not change the value of covariance [20]. Therefore, it does not affect the convergence of the ITUKF. In the following analysis, we only analyze the convergence of the improved TUKF combined with the proposed fuzzy noise estimator. The process and measurement functions are given by

$$X_k = f(X_{k-1}) + w_k \tag{12}$$

$$Z_k = h(X_k) + v_k \tag{13}$$

where X_k is the state vector at time k, Z_k is the measurement vector, w_k and v_k are the process and measurement noise with zero-mean and covariance of $\mathbf{Q}$ and $\mathbf{R}$, $\mathrm{f}(\cdot)$ denotes the process/state function, and $h(\cdot)$ denotes the measurement function. Define two errors for standard UKF as follows:

$$\tilde{X}_k = X_k - \hat{X}_k \tag{14}$$

$$\tilde{X}_{k|k-1} = X_k - \hat{X}_{k|k-1} \tag{15}$$

where $\tilde{X}_k$ and $\tilde{X}_{k|k-1}$ denote the update and prediction errors, respectively. Combing the prediction process in the conventional UKF [24] and (14), and expanding $\mathrm{f}(\cdot)$ by a Taylor series about $\hat{X}_{k-1}$, the prediction error can be rewritten as

$$\tilde{X}_{k|k-1} = \mathbf{F}_k \tilde{X}_{k-1} + \Delta(\tilde{X}_{k-1}) + w_k \tag{16}$$

where $\mathbf{F}_k$ is the Jacobian matrix of $\mathrm{f}(\cdot)$ and $\Delta(\tilde{X}_{k-1})$ represents the second and higher-order moments in the Taylor series. To simplify (18), an unknown time-varying diagonal matrix $\boldsymbol{\beta}_k = \mathrm{diag}(\beta_{1,k}, \beta_{2,k}, \ldots, \beta_{n,k})$ is used [24], and it works by the first-order linearization. Therefore, (18) can be rewritten as follows:

$$\tilde{X}_{k|k-1} = \boldsymbol{\beta}_k \mathbf{F}_k \tilde{X}_{k-1} + w_k. \tag{17}$$

According to (19) and [13], the true prediction covariance can be rewritten as follows:

$$\begin{aligned} \mathbf{P}^r_{k|k-1} &= E\left[\tilde{X}_{k|k-1}\tilde{X}^T_{k|k-1}\right] \\ &= \boldsymbol{\beta}_k \mathbf{F}_k \mathbf{P}_{k-1} \mathbf{F}_k^T \boldsymbol{\beta}_k^T + \Delta \mathbf{P}_{k|k-1} + \mathbf{Q} \end{aligned} \tag{18}$$

where $\Delta \mathbf{P}_{k|k-1} = E\left[\boldsymbol{\beta}_k \mathbf{F}_k \tilde{X}_{k-1} \tilde{X}^T_{k-1} \mathbf{F}_k^T \boldsymbol{\beta}_k^T\right] - \boldsymbol{\beta}_k \mathbf{F}_k \mathbf{P}_{k-1} \mathbf{F}_k^T \boldsymbol{\beta}_k^T$. The calculated prediction covariance is given by

$$\begin{aligned} \mathbf{P}_{k|k-1} &= \boldsymbol{\beta}_k \mathbf{F}_k \mathbf{P}_{k-1} \mathbf{F}_k^T \boldsymbol{\beta}_k^T + \Delta \mathbf{P}_{k|k-1} + \delta \mathbf{P}_{k|k-1} + \mathbf{Q} \\ &= \boldsymbol{\beta}_k \mathbf{F}_k \mathbf{P}_{k-1} \mathbf{F}_k^T \boldsymbol{\beta}_k^T + \tilde{\mathbf{Q}} \end{aligned} \tag{19}$$

where $\delta \mathbf{P}_{k|k-1} = \mathbf{P}_{k|k-1} - \mathbf{P}^r_{k|k-1}$ and $\tilde{\mathbf{Q}} = \Delta \mathbf{P}_{k|k-1} + \delta \mathbf{P}_{k|k-1} + \mathbf{Q}$.

Lemma 1: With the process and measurement functions given in (14) and (15), the estimation error of the standard UKF is bounded in mean square and converged if the following assumptions hold for each time step [24]:

$$f^2_{\min}\mathbf{I} \le \mathbf{F}_k \mathbf{F}_k^T \le f^2_{\max}\mathbf{I} \tag{20}$$

$$h^2_{\min}\mathbf{I} \le \mathbf{H}_k \mathbf{H}_k^T \le h^2_{\max}\mathbf{I} \tag{21}$$

$$\beta^2_{\min}\mathbf{I} \le \boldsymbol{\beta}_k \boldsymbol{\beta}_k^T \le \beta^2_{\max}\mathbf{I} \tag{22}$$

$$\mathbf{Q} \le q_{\max}\mathbf{I} \tag{23}$$

$$\mathbf{R} \le r_{\min}\mathbf{I} \tag{24}$$

$$\tilde{q}_{\max}\mathbf{I} \ge \tilde{\mathbf{Q}} \ge \tilde{q}_{\min}\mathbf{I} \tag{25}$$

$$p_{\min}\mathbf{I} \ge \mathbf{P}_k \ge p_{\max}\mathbf{I} \tag{26}$$

where $f_{\min}$, $h_{\min}$, $\beta_{\min}$,$f_{\max}$, $h_{\max}$ and $\beta_{\max}$ are nonzero real numbers, $q_{\max}$, $\tilde{q}_{\max}$, $q_{\max}$,$r_{\min}$, $p_{\max}$ and $p_{\min}$ are positive real numbers, $\mathbf{I}$ denotes the identity matrix, and $\mathbf{H}_k$ denotes the Jacobian matrix of measurement function. When the process or measurement noise is inaccurate, condition (25) may not be satisfied. Even if (25) is satisfied, the filter is also biased from the minimum covariance [13]. To solve this problem, the proposed FNE provides a possible solution to improve the stability of the filter under unknown/inaccurate noise statistics.

Theorem 1: If $\Delta \mathbf{Q}_k$ and $\Delta \mathbf{R}_k$ are two positive definite matrixes, the convergence of the proposed ITUKF can be achieved by adding an appropriate $\Delta \mathbf{Q}_k$ and $\Delta \mathbf{R}_k$ to the modified covariance of process and measurement noises, respectively.

Proof: Similar to (19), the prediction covariance of ITUKF can be expressed as

$$\mathbf{P}_{k|k-1} = \boldsymbol{\beta}_k \mathbf{F}_k \mathbf{P}_{k-1} \mathbf{F}_k^T \boldsymbol{\beta}_k^T + \tilde{\mathbf{Q}}^* \tag{27}$$

where $\tilde{\mathbf{Q}}^* = \Delta\mathbf{P}_{k|k-1} + \delta\mathbf{P}_{k|k-1} + \hat{\mathbf{Q}}^*$. To achieve the convergence of the ITUKF, an appropriate positive definite matrix $\Delta\mathbf{Q}_k$ should be added to $\tilde{\mathbf{Q}}^*$ such that

$$\tilde{\mathbf{Q}}^{**} = \tilde{\mathbf{Q}}^* + \Delta\mathbf{Q}_k = \Delta\mathbf{P}_{k|k-1} + \delta\mathbf{P}_{k|k-1} + \mathbf{Q} + \Delta\mathbf{Q}_k > \tilde{\mathbf{Q}}. \tag{28}$$

If $\Delta\mathbf{Q}_k$ is selected appropriately, the condition (28) can be achieved by substituting (28) to (25) of Lemma 1. Similar to the proof of process noise, the condition (26) can be achieved using modified measurement noise. This reveals that the convergence/stability of the ITUKF can be achieved by adding an appropriate $\Delta\mathbf{Q}_k$ and $\Delta\mathbf{R}_k$ to the estimated $\hat{\mathbf{Q}}^*$ and $\hat{\mathbf{R}}^*$. The proof of Theorem 1 is completed.

IV. ITUFastSLAM

This section provides the implementation of the proposed algorithm, which contains the importance sampling, feature update, calculation of the importance weights, and the adaptive GA resampling. Finally, a system diagram is concluded.

A. Importance Sampling

The first step for executing the ITUFastSLAM is to construct the augmented state vector by adding the mean and covariance of the process noise. If the mean of the process noise is zero, the augmented state is provided as follows:

$$s_{t-1}^{a[m]} = \begin{bmatrix} s_{t-1}^{[m]} \\ 0 \end{bmatrix}, P_{t-1}^{a[m]} = \begin{bmatrix} \mathbf{P}_{t-1}^{[m]} & 0 \\ 0 & \mathbf{Q}_{t-1} \end{bmatrix} \tag{29}$$

where $s_{t-1}^{a[m]}$ is the augmented vector, $\mathbf{Q}_{t-1}$ is the process noise variance, and $s_{t-1}^{[m]}$ and $\mathbf{P}_{t-1}^{[m]}$ are the mean and covariance of the vehicle state at the last time step, respectively. The dimension of $s_{t-1}^{a[m]}$ is n_a. Use the Cholesky factorization to decompose the matrix $P_{t-1}^{a[m]}$ as follows:

$$\mathbf{M}_{t-1}^{a[m]} = \text{chol}\left(\mathbf{P}_{t-1}^{a[m]}\right) \tag{30}$$

where the operator $\text{chol}(\cdot)$ 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 $2n_a$ *sigma points* are calculated as follows:

$$\chi_{t-1}^{a[i][m]} = s_{t-1}^{a[m]} + \left(\mathbf{M}_{t-1}^{a[m]}\right)^T \zeta^{[i]}, \qquad i = 1, 2, \ldots, 2n_a \tag{31}$$

where $\zeta^{[i]}$ is the ith vector as introduced in (6)–(8). Each *sigma point* $\chi_{t-1}^{a[i][m]}$ contains the state and the process noise components that given by

$$\chi_{t-1}^{a[i][m]} = \begin{bmatrix} \chi_{t-1}^{[i][m]} \\ \chi_t^{u[i][m]} \end{bmatrix}. \tag{32}$$

The set of *sigma points* are propagated through the nonlinear motion model as follows:

$$\bar{\chi}_{t|t-1}^{[i][m]} = f\left(u_t + \chi_t^{u[i][m]}, \chi_{t-1}^{[i][m]}\right). \tag{33}$$

Here, $\bar{\chi}_{t|t-1}^{[i][m]}$ is the transformed *sigma point* of the state. The predicted mean of the state is calculated based on the weights as

$$s_{t|t-1}^{[m]} = \sum_{i=1}^{2n_a} W\bar{\chi}_{t|t-1}^{[i][m]} \tag{34}$$

where $W = 1/(2n_a)$. Executing the Cholesky factorization is computationally expensive because it contains a set of weighted deviation e_i as shown in (35). Hence, the QR decomposition is performed on the matrix $A = [e_1 e_2 \ldots e_{2n_a}]$ to reduce the computational efforts [20]

$$e_i = \sqrt{W}\left(\bar{\chi}_t^{[i][m]} - s_{t|t-1}^{[m]}\right), i = 1, \ldots, 2n_a \tag{35}$$

$$\bar{\mathbf{M}}_{t|t-1}^{[m]} = qr(A) = qr[e_1\ e_2\ \ldots\ e_{2n_a}] \tag{36}$$

where $\bar{\mathrm{M}}_{\mathrm{t|t-1}}^{[\mathrm{m}]}$ 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:

$$\bar{\gamma}_t^{[i][m]} = h\left(\bar{\chi}_{t|t-1}^{[i][m]}, \mu_{t-1}^{[m]}\right) \tag{37}$$

$$\hat{z}_t^{[m]} = \sum_{i=1}^{2n_a} W\bar{\gamma}_t^{[i][m]}. \tag{38}$$

The weighted deviations

$$\varsigma_i = \sqrt{W}\left(\bar{\gamma}_t^{[i][m]} - \hat{z}_t^{[m]}\right), \qquad i = 1, \ldots, 2n_a \tag{39}$$

are used to derive

$$\bar{\mathbf{S}}_{z_t}^{[m]} = qr\left[\varsigma_1\ \varsigma_2\ \ldots\ \varsigma_{2n_a}\ \sqrt{\mathbf{R}_{t-1}}\right] \tag{40}$$

where $\mathbf{R}_{t-1}$ is the measurement noise variance, and it is considered as an additive term instead of the augmented term [4]. $\bar{\gamma}_t^{[i][m]}$ is the propagated measurement by ITUKF where $\mathrm{h}(\cdot)$ denotes the measurement function. $\bar{\mathbf{S}}_{z_t}^{[m]}$ is the upper triangular matrix of the covariance matrix of $\hat{z}_t^{[m]}$. For adjusting the noises, the residual of the observation is first calculated

$$r_t = z_t - \hat{z}_t^{[m]}. \tag{41}$$

The average residual $\bar{r}_t$ and its covariance $\bar{\mathbf{P}}_{r,t}$ are then provided by

$$\bar{r}_t = \frac{1}{q}\sum_{i=1}^{q} r_{t+i-q}^{[m]} \tag{42}$$

$$\bar{\mathbf{P}}_{r,t} = \frac{1}{q}\sum_{i=1}^{q} r_{t+i-q}^{[m]}\left(r_{t+i-q}^{[m]}\right)^T \tag{43}$$

where q is the length of the time window. The process noise and measurement noise can be adjusted as

$$\mathbf{R}_t = (1+\lambda)\mathbf{R}_{t-1}, \quad \mathbf{Q}_t = (1+\lambda)\mathbf{Q}_{t-1}. \tag{44}$$

According to the adjusted noises, the square root covariance $\bar{\mathbf{S}}_{z_t}^{[m]}$ should be recalculated as (40), and the modified process noise is propagated to the next period.

The cross covariance is found just as in the UKF

$$\mathbf{P}_{s_t z_t}^{[m]} = \sum_{i=1}^{2n_a} W \left(\bar{\chi}_t^{[i][m]} - s_{t|t-1}^{[m]}\right)\left(\bar{\gamma}_t^{[i][m]} - \hat{z}_t^{[m]}\right)^T. \quad (45)$$

The Kalman gain matrix and measurement update are given by

$$\mathbf{K}_t^{[m]} = \left(\mathbf{P}_{s_t z_t}^{[m]} / \bar{\mathbf{S}}_{z_t}^{[m]}\right) \Big/ \left(\bar{\mathbf{S}}_{z_t}^{[m]}\right)^T \quad (46)$$

$$\hat{s}_t^{[m]} = s_{t|t-1}^{[m]} + \mathbf{K}_t^{[m]}\left(z_t - \hat{z}_t^{[m]}\right) \quad (47)$$

$$\mathbf{U} = \mathbf{K}_t^{[m]}\left(\bar{\mathbf{S}}_{z_t}^{[m]}\right)^T \quad (48)$$

$$\mathbf{M}_t^{[m]} = \text{cholupdate}\left\{\mathbf{M}_{t|t-1}^{[m]}, \mathbf{U}, -1\right\} \quad (49)$$

$$\mathbf{P}_t^{[m]} = \left(\mathbf{M}_t^{[m]}\right)^T \mathbf{M}_t^{[m]} \quad (50)$$

where z_t is the observation input, $s_t^{[m]}$ is the state mean, and $M_t^{[m]}$ is the upper triangular decomposition of the covariance matrix. The function cholupdate $(\cdot)$ transforms the Cholesky decomposition of a matrix $\mathbf{A}$ into the Cholesky decomposition of the matrix $\mathrm{A} \pm XX^T$ where X is a column vector. For the Gaussian distribution, the state of each particle is sampled by

$$s_t^{[m]} \sim N\left(\hat{s}_t^{[m]}, \mathbf{P}_t^{[m]}\right). \quad (51)$$

B. Feature Update

If a landmark n_t is revisited at time t, then the *sigma points* are defined using the previously registered mean $\mu_{n_t,t-1}^{[m]}$ and covariance $\sum_{n_t,t-1}^{[m]}$ as follows:

$$\delta^{[i][m]} = \mu_{n_t,t-1}^{[m]} + (\mathbf{\Omega}_{n_t,t-1}^{[m]})^T \zeta^{[i]}, \qquad i = 1, \ldots, 2n \quad (52)$$

where $\mathbf{\Omega}_{n_t,t-1}^{[m]}$ is the upper triangular decomposition of the feature covariance matrix, and n is the dimension of the feature state. If landmarks are on a planar environment, $n = 2$. The *sigma points* are propagated through the measurement model as follows:

$$\bar{Z}_t^{[i][m]} = h\left(\delta^{[i][m]}, s_t^{[m]}\right) \quad (53)$$

$$\hat{\Gamma}_t^{[m]} = \sum_{i=1}^{2n} W_n \bar{Z}_t^{[i][m]} \quad (54)$$

$$\tau_i = \sqrt{W_n}\left(\bar{Z}_t^{[i][m]} - \hat{\Gamma}_t^{[m]}\right), \qquad i = 1, \ldots, 2n \quad (55)$$

$$\bar{\mathbf{S}}_{\Gamma_t}^{[m]} = qr\left[\tau_1 \ \tau_2 \ \ldots \ \tau_{2n} \ \sqrt{\mathbf{R}_t}\right] \quad (56)$$

where $s_t^{[m]}$ is the current robot state of the mth particle and $W_n = 1/(2n)$.

The cross-covariance $\bar{\mathbf{\Lambda}}_{z_t s_t}^{[m]}$ and the Kalman gain are calculated by

$$\bar{\mathbf{\Lambda}}_{z_t s_t}^{[m]} = \sum_{i=1}^{2n} W_n \left(\delta^{[i][m]} - \mu_{n_t,t-1}^{[m]}\right)\left(\bar{Z}_t^{[i][m]} - \hat{\Gamma}_t^{[m]}\right)^T \quad (57)$$

$$\bar{\mathbf{K}}_t^{[m]} = \left(\bar{\mathbf{\Lambda}}_{z_t s_t}^{[m]} \Big/ \bar{\mathbf{S}}_{\Gamma_t}^{[m]}\right) \Big/ \left(\bar{\mathbf{S}}_{\Gamma_t}^{[m]}\right)^T. \quad (58)$$

The mean and the square root of covariance of the feature are updated as

$$\mu_{n_t,t}^{[m]} = \mu_{n_t,t-1}^{[m]} + \bar{\mathbf{K}}_t^{[m]}\left(z_t - \hat{\Gamma}_t^{[m]}\right) \quad (59)$$

$$\mathbf{U} = \bar{\mathbf{K}}_t^{[m]}\left(\bar{\mathbf{S}}_{\Gamma_t}^{[m]}\right)^T \quad (60)$$

$$\mathbf{\Omega}_t^{[m]} = \text{cholupdate}\left\{\mathbf{\Omega}_{t-1}^{[m]}, \mathbf{U}, -1\right\}. \quad (61)$$

C. Calculating the Importance Weights

Like the UFastSLAM, the importance weight should consider the latest observation, and it is given by [17]

$$w_t^{[m]} = w_{t-1}^{[m]} \frac{p\left(z_t | \hat{s}_t^{[m]}\right) p\left(s_t^{[m]} | s_{t-1}^{[m]}, u_t\right)}{N\left(\hat{s}_t^{[m]}, \mathbf{P}_t^{[m]}\right)} \quad (62)$$

where $\mathrm{N}(\cdot)$ denotes the Gaussian distribution. Equation (62) comes from (5), and the detailed derivation can refer to [17], and 25]. A general problem of PF is the degraded particles that have small weights. The degeneracy degree of the PF is judged by the effective particle number N_{eff} provided by [17]

$$N_{\text{eff}} = \frac{1}{\sum_{m=1}^{M}\left(\hat{w}_t^{[m]}\right)^2} \quad (63)$$

where M is the number of particles and $\hat{w}_t^{[m]}$ is the normalized weight of the mth particle. If the effective number is smaller than the preset threshold, the resampling step will be executed to mitigate the degeneracy of the PF. Hopefully, the filter will obtain the independent identically distributed particles with the same weight after resampling.

D. Adaptive GA Resampling

In this section, a well-performed GA is first introduced [25], which works by merging the large-weight and small-weight particles during each period. Compared with the GA optimization, the proposed resampling is only carried out when the effective particle number is less than the specified threshold. Moreover, the coefficients of two fundamental GA operators, crossover and mutation, are provided according to the distribution of particle weights.

1) GA Optimization: The particle set $\{s_t^{[m]}, \hat{w}_t^{[m]}\}$ is first divided into two parts with a separating threshold W_T based on the importance weights. Sort the particle weights $\hat{w}_t^{[m]} (m = 1, \ldots, M)$ in a descending order as $\Theta = \{\hat{w}_t^{[1]}, \hat{w}_t^{[2]}, \ldots, \hat{w}_t^{[M]}\}$ where $\Theta(m)$ represents the mth weight in the set Θ. The

Algorithm 1: ITUFastSLAM.

Input: $s_0, \mathbf{P}_0, z_t, M, \mathbf{R}_0, \mathbf{Q}_0$
Result: $s_t, \mu_{n_t,t}$

Initialize particle weights to $1/M$
STEP1:
for $i = 1 : M$
Importance sampling by ITUKF:
1) Predict the vehicle state using (29)–(36).
2) Update the vehicle state using (37)–(51) where the modified noises (44) are calculated according to the fuzzy rule provided by Table I and Fig. 1, and (40) should be recalculated to update the state.
Feature Update:
3) Update the location of features using (52)–(61).
end for
STEP2: *Calculate the particle weights*
1) Calculate the particle weights using (62) and normalize weights, and then calculate the effective number of particles using (63).
if $\mathrm{N_{eff}}$ <threshold, go ***STEP3***.
else reset particle weights to 1/*M* and calculate the weighted mean of vehicle and features states; go ***STEP1***.
end if
STEP3: *Perform adaptive genetic resampling*
1) Use crossover and mutation operators (66), (67) where the crossover coefficients and mutation probability are given by (68) and (69) to modify the vehicle state.
2) Reset particle weights to 1/*M* and calculate the weighted mean of vehicle and features states; go ***STEP1***.

segmentation threshold W_T is given by

$$W_T = \Theta\left(\bar{N}_{\mathrm{eff}}\right) \tag{64}$$

where $\bar{N}_{\mathrm{eff}} = \mathrm{round}\left[N_{\mathrm{eff}}\right]$ and the operator $\mathrm{round}\left[\cdot\right]$ denotes the rounding symbol. The divided particles can be consequently expressed as

$$s_t^{[m]} \in \begin{cases} C_L, & \hat{w}_t^{[m]} \le W_T \\ C_H, & \hat{w}_t^{[m]} > W_T \end{cases} \tag{65}$$

where C_L and C_H denote the small-weight and large-weight particle sets, respectively. Assume that $s_{t,L}^l$ and $s_{t,H}^l$ represent the particles from C_L and C_H, respectively. If $s_{t,C}^l$ represents the modified particles, the formulation of the crossover operator is given by

$$s_{t,C}^l = \eta s_{t,L}^l + (1-\eta) s_{t,H}^j \tag{66}$$

in which $l = 1, \ldots, N_L$ and $j = 1, \ldots, N_H$. N_L and N_H denote the numbers of particles contained in C_L and C_H, respectively. For each $s_{t,C}^l$, $s_{t,H}^l$ is randomly selected from the set C_H. The parameter $\eta \in [0, 1]$ represents the crossover degree of the particles. A mutation operator is performed to further promote the diversity of the particles. This step may happen on the modified particles $s_{t,C}^l$ according to a probability as

$$s_{t,M}^l = \begin{cases} 2s_{t,H}^j - s_{t,C}^l, & r_l \le p_M \\ s_{t,C}^l, & r_l > p_M \end{cases} \tag{67}$$

where r_l is a random variable drawn from the uniform distribution on [0,1]. p_M is the mutation probability, and $s_{t,M}^l$ is the mutated particle set. It is noted that the particle is not only modified by the parent particles but also by the same generation.

2) Adaptive Coefficients: Generally, the parameters of the crossover degree η and mutation probability p_M are tuned according to different systems. Here, from the perspective of probability, an adaptive selection method for these two parameters is proposed. For a set of numbers (normalized weights) varied within [0,1], its variance is within the range of [0,1/4] [26]. It is approved that the smaller the variance of particle weights, the better the particles approximate the true posterior distribution. Hence, according to the variance σ^2 of the particle weights, a linear function is used to decide the relationship between η and σ^2, and it is given by

$$\eta = 1 - 4\sigma^2. \tag{68}$$

When σ^2 is equal to 0, the parameter $\eta = 1$, which means no crossover occurs. On the contrary, when σ^2 is equal to 1/4, η is 0, which means the greater degree crossover is needed to mitigate the degeneracy and impoverishment problems. The mutation probability p_M is given by

$$p_M = \begin{cases} p_{\max} - \frac{(p_{\max} - p_{\min})(\bar{w}_t^{[m]} - \hat{w}_{t,\mathrm{avg}}^{[m]})}{\hat{w}_{t,\max}^{[m]} - \hat{w}_{t,\mathrm{avg}}^{[m]}}, & \bar{w}_t^{[m]} > \hat{w}_{t,\mathrm{avg}}^{[m]} \\ p_{\max}, & \bar{w}_t^{[m]} \le \hat{w}_{t,\mathrm{avg}}^{[m]} \end{cases} \tag{69}$$

where $p_{\max}$ and $p_{\min}$ denote the maximum and the minimum mutation probability, respectively. Generally, $p_{\min} = 0.005$ and $p_{\max} = 0.01$ [27]. $\hat{w}_{t,\max}^{[m]}$ and $\hat{w}_{t,\mathrm{avg}}^{[m]}$ denote the maximum and average weights ranked in the set W, respectively. $\bar{w}_t^{[m]}$ denotes the weight of the modified particle $x_{t,C}^l$. When $\bar{w}_t^{[m]} > \hat{w}_{t,\mathrm{avg}}^{[m]}$, a lower mutation probability is preferred because the modified particle tends to a higher posterior probability region compared with the particle which has the weight of $\bar{w}_t^{[m]} \le \hat{w}_{t,\mathrm{avg}}^{[m]}$. When $\bar{w}_t^{[m]}$ is lower than $\hat{w}_{t,\mathrm{avg}}^{[m]}$, a higher mutation probability $p_{\max}$ is suggested to overcome the impoverishment problem of the particles. Finally, the particles can be updated and their weights should be initialized.

The implementation of the proposed algorithm is demonstrated in Algorithm 1.

V. Results of Simulation and Experiment

The proposed algorithm is first evaluated in the simulator provided by Bailey [21], and then verified with the real-world datasets of Car Park and Victoria Park, respectively [22]. To better demonstrate the performance of the proposed method, it is compared with the unscented FastSLAM (UFastSLAM) [4] and the transformed unscented FastSLAM (TUFastSLAM), which employs the TUKF as importance sampling function with a systematic resampling.

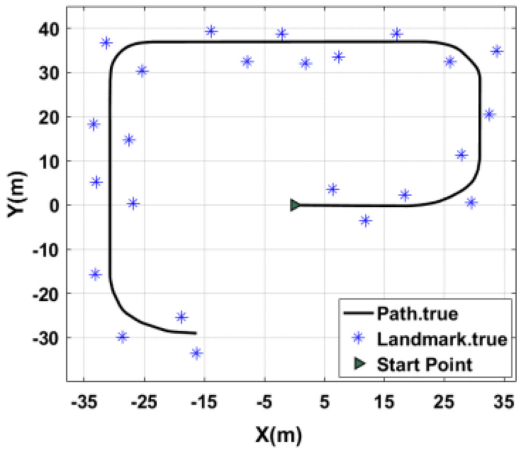

Fig. 3. True trajectory and landmarks in the simulator. (control noise: $\sigma_v = 0.3$ m/s, $\sigma_g = 3.0^\circ$).

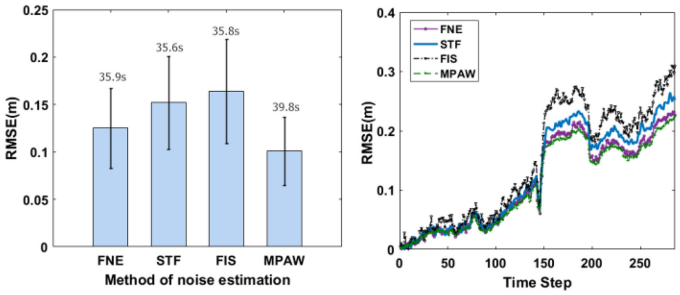

Fig. 4. RMSE and cost time of different method for noise adjustment. (particle number: 10, measurement noise: $\sigma_\rho = 0.8$ m, $\sigma_\theta = 3.0^\circ$).

A. Simulation Results

In the simulator, the robot moves at a speed of 3 m/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 kinematic and measurement models are provided in [4] where the location and azimuthal angle of the vehicle are the state variables, and the distance and relative azimuthal angle of the observed feature are the measurement variables. Fig. 3 demonstrates the true trajectory of the robot and landmarks location of the test environment. The control and observation frequencies are set at 40 Hz and 5 Hz, respectively. For each test, the results are obtained over 30 Monte Carlo runs, and the data association is assumed known over the entire process. The length of time window given in (42) and (43) is set to 3.

1) Performance of the Proposed Fuzzy Noise Estimator: This part demonstrates the performance of the proposed FNE compared with the STF [5], FIS [12] and MPAW [13], respectively. Fig. 4 demonstrates the root-mean-square error (RMSE) and consuming time of four algorithms of noise adjustment. Compared with the STF and FIS, the proposed FNE spends nearly the same time but has a smaller RMSE by 26.7% and 31.3%, respectively. This is because the FNE not only considers the effect of residual but also its related covariance. In contrast to the MPAW, the FNE reduce the consuming time by 9.8% since it can be easily operated by referring to the fuzzy rule table instead of complicated iteration of maximum posterior estimation [13]. Above all, the proposed FNE is suitable for the real-time applications in which the estimation accuracy should also meet the specified requirement.

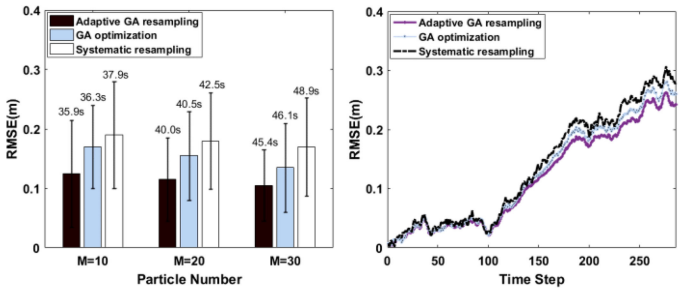

Fig. 5. Performance analysis of adaptive genetic resampling. (Measurement noise: $\sigma_\rho = 0.8$ m, $\sigma_\theta = 3.0^\circ$. The right picture is acquired with 10 particles).

2) Performance of Adaptive Genetic Resampling: To better highlight the performance, the adaptive genetic resampling is tested compared with the GA optimization and ($\eta = 0.1$, $p_M = 0.5$) [25] and systematic resampling [15]. The RMSE and the total cost time of the proposed method are presented in Fig. 5, respectively. It can be seen that using the adaptive genetic resampling can help improve the estimation accuracy compared with that with GA optimization by 29.4%, 28.6%, and 14.3% with 10, 20, and 30 particles, respectively. This is because the crossover and mutation coefficients, which should be empirically tuned in GA optimization, are calculated according to the distribution of weights of particles. In this way, the proposed resampling can better approximate the true state of the vehicle. Moreover, the proposed resampling can also reduce the computational time by 1.1%, 1.2%, and 1.5% when $M = 10$, 20, and 30, respectively, since the proposed resampling is only carried out when the effective particles are less than the threshold. Compared with the systematic resampling, the proposed 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.

3) Performance Comparison With Different Measurement Noise and Number of Particles: To test the performance of the proposed algorithm, the preset condition is given under different noise levels and the number of particles as shown in Fig. 6, respectively. Fig. 6(a) and (b) illustrate the corresponding RMSE of the estimated vehicle position and heading angle. It is obvious that the mean and variance of RMSE increase as the measurement noises increase for different algorithms. Compared with the TUFastSLAM, the proposed method increases the estimation accuracy of vehicle position by 38.9%, 37.5%, and 33.8% under measurement noise level A, B, and C, respectively, because of the better proposal distribution and resampling. For a different number of particles ($M = 10$, 20, and 30), the proposed ITUFastSLAM also presents better accuracy

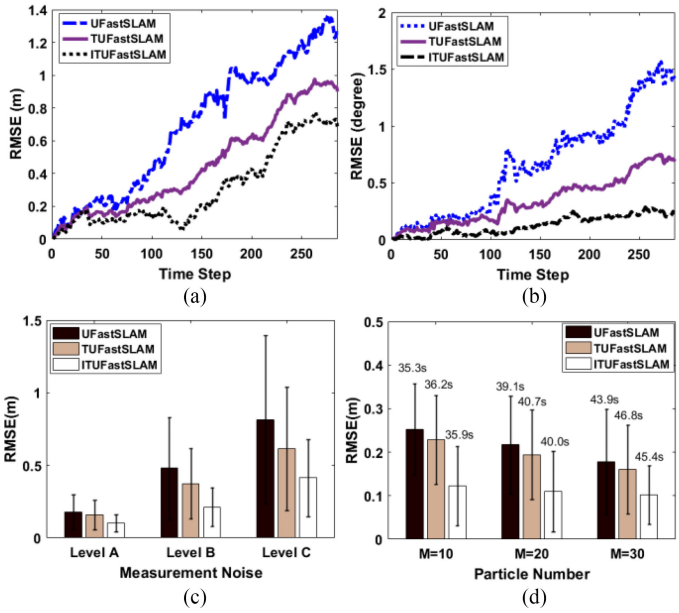

Fig. 6. (a) RMSE of the vehicle position under three measurement noise levels with 30 particles. (b) RMSE of the vehicle position with different number of particles under level A. (c) and (d) RMSE curves of vehicle position and heading angle under measurement noise level A with 10 particles, respectively. (Level A: $\sigma_\rho = 0.1$ m, $\sigma_\theta = 0.5^\circ$, Level B: $\sigma_\rho = 0.5$ m, $\sigma_\theta = 2.0^\circ$, and Level C: $\sigma_\rho = 0.8$ m, $\sigma_\theta = 3.0^\circ$).

TABLE III
COMPOSITION OF THREE ALGORITHMS

Composition	**Importance sampling function**	**Resampling**
UFastSLAM	UKF	Systematic resampling
TUFastSLAM	TUKF	Systematic resampling
ITUFastSLAM	TUKF with square root filter and fuzzy noise estimator	Adaptive GA resampling

than the TUFastSLAM and UFastSLAM. It can be seen that the ITUFastSLAM does not heavily depend on the number of particles, and its accuracy increases by 9.3% and 8.1% when the number of particles varies from 10 to 30. This is because the adaptive GA resampling makes the particles tend to high posterior probability region of the true state with no need for numerous particles.

4) Computational Cost: The computational cost of the proposed algorithm is analyzed using MATLAB simulations on Intel(R) Core(TM) i5-4590 CPU@3.3GHz PC. The CPU running time of each SLAM algorithm is utilized to evaluate the computational complexity. Table III demonstrates the composition of three algorithms. As shown in Fig. 6(d), the CPU running time of three algorithms increases as the number of particles increases. Under the specified particle number, ITUFastSLAM has better computational efficiency compared with the TUFastSLAM. When the number of particles varies from 10 to 30, the proposed method reduces the CPU running time by 0.8%, 1.7%, and 3.0%, respectively. This result is a combined

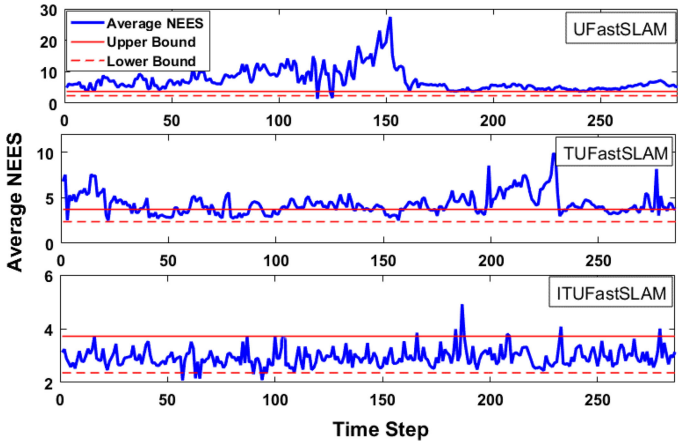

Fig. 7. Average NEES for UFastSLAM, TUFastSLAM, and the proposed method. (Measurement noise: Level A; particle number: 30).

effect of the submodules in ITUFastSLAM. The square root filter propagates the covariance with QR decomposition and reduces the computational complexity [5], [17]. Moreover, the adaptive GA resampling only conducts on small-weight particles rather than all particles compared with the systematic resampling [15]. In this way, the computational efficiency can be improved. Although the FNE adds additional steps to the proposed algorithm, the combined effect of the submodules makes ITUFastSLAM have an improved computational efficiency than TUFastSLAM. Combining the results in Section V-A-3), the proposed algorithm has better accuracy and computational efficiency than TUFastSLAM.

5) Filter Consistency of the Proposed Method: To verify the consistency of the filter, the normalized estimation error squared (NEES) is used [5], [17], and it is given by

$$\varepsilon_t = (s_t - \hat{s}_t)^T \mathbf{P}_t^{-1} (s_t - \hat{s}_t) \tag{70}$$

where s_t, $\hat{s}_t$ and P_t represent the ground truth, estimated mean, and the covariance of the robot trajectory, respectively. Consistency is evaluated by performing multiple Monte Carlo runs and computing the average NEES. Given N_R runs, the average NEES (ANESS) is computed as follows:

$$\bar{\varepsilon}_t = \frac{1}{N_R} \sum_{i=1}^{N_R} \varepsilon_{\text{it}}. \tag{71}$$

For the three-dimensional (3-D) robot pose and 30 Monte Carlo runs, the 95%-confidence level is bounded by interval [2.36, 3.72]. The average ANEES of UFastSLAM, TUFastSLAM, and ITUFastSLAM is 9.3, 5.0, and 2.9 respectively; therefore, the average ANEES of the proposed method is reduced by 42% and 69% than the TUFastSLAM and UFastSLAM, respectively. The distribution of ANEES is shown in Fig. 7. In general, the consistency of the proposed method is better than the UFastSLAM and the TUFastSLAM since it almost locates within the preset confidence interval. This is because the better proposal distribution is used in the algorithm than that of the UFastSLAM and TUFastSLAM. Furthermore, the intelligent GA resampling makes the particles

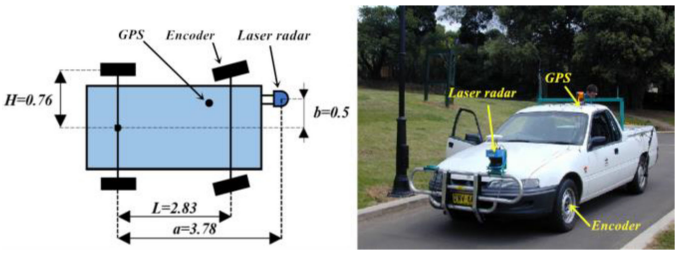

Fig. 8. Basic parameters of the mobile vehicle.

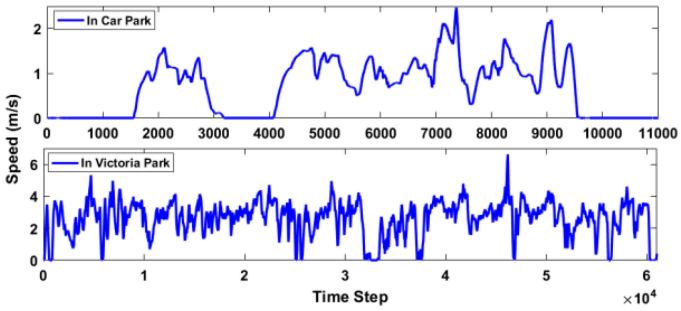

Fig. 9. Speed of vehicle in Car Park and Victoria Park.

well-approximate the posterior distribution of the true state and maintains the diversity of the particles.

B. Experimental Results

The proposed algorithm is compared with the UFastSLAM and the TUFastSLAM using the popular datasets collected in Car Park and Victoria Park, respectively. The experimental platform was a four-wheeled vehicle equipped with the wheel encoders (ROD-430), GPS (Ashtech GG24), and a laser radar (SICK LMS 221). The wheel encoders are used to measure the steering angle and velocity of the vehicle. The corresponding standard deviations of the encoders are $\pi/30$ rad and 2 m/s, respectively. The GPS provides the ground true of the vehicle, and it is used for verifying the performance of the proposed SLAM algorithm. The laser radar, which has a 180° frontal field-of-view, is used to perceive the range and bearing angle of the environmental features with the standard deviation of 1 m and $\pi/60$ rad, respectively. The steel poles and trees are the perceived features of laser radar in Car Park and Victoria Park, respectively. The geometric parameters and equipped sensors of the car are shown in Fig. 8. The speed of the vehicle is shown in Fig. 9. The Ackerman model is selected as the process function for the proposed algorithm [22]. Moreover, the individual compatibility nearest neighbor test with a 2σ acceptance region is used for data association of features.

In the two tests, the uneven terrain caused the wheel slippage and attitude errors, and thus induced poor odometry. Some man-made landmarks consisted of 60-mm steel poles covered with reflective tape were used to improve the observing accuracy in the Car Park. The vehicle was driven around the park for about 30 min and travels over 4 km in the Victoria Park. Due to the occlusion by foliage and buildings, the ground-true

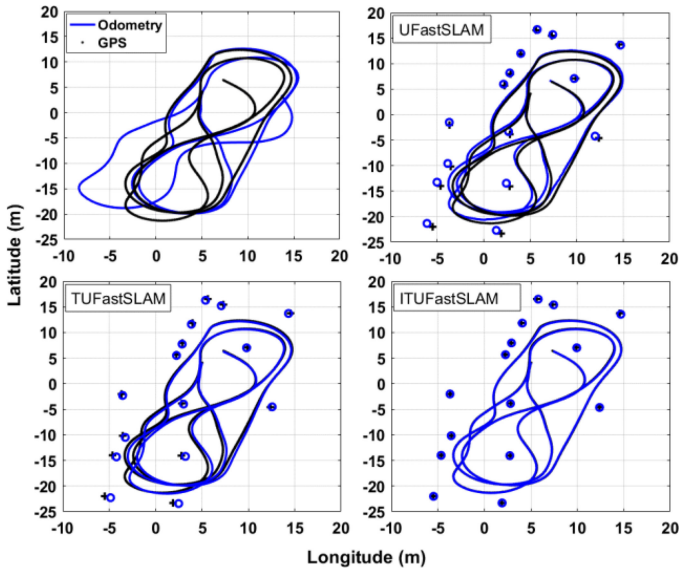

Fig. 10. Comparison of GPS and estimated paths in Car Park. (Particle number: 10). The black and blue lines denote the GPS and estimated path of the vehicle, and "+" and "o" denote the true and estimated positions of landmarks.

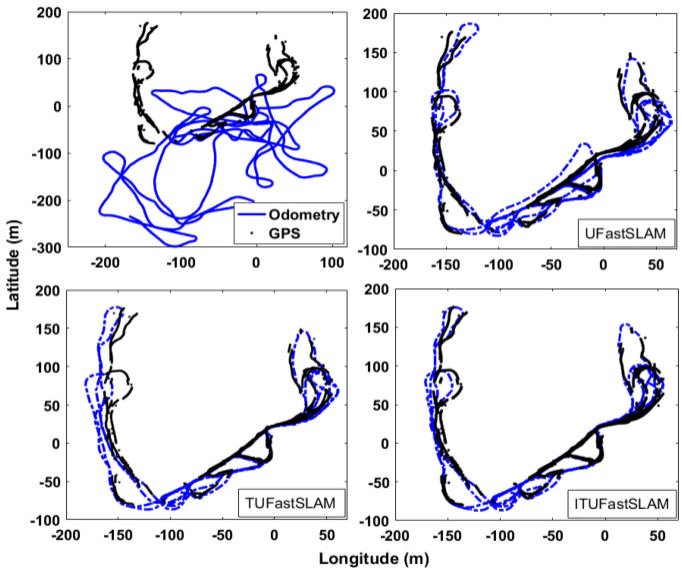

Fig. 11. Comparison of the GPS path and the estimated paths of UFastSLAM, TUFastSLAM, and the ITUFastSLAM with 10 particles. The black and blue lines denote the GPS and estimated path of the vehicle.

position of the vehicle was not available throughout the experiment but enough to verify the positioning accuracy. Figs. 10 and 11 demonstrate the GPS and estimated vehicle paths in Car Park and Victoria Park, respectively. Because of the randomness in importance sampling by (51), different vehicle paths will be acquired in each test even if the test condition is unchanged. Therefore, the superiority of the proposed algorithm should be verified using a certain number of Monte Carlo tests. Table IV provides the RMSE of the vehicle position and running time of the proposed algorithm compared with the TUFastSLAM and

TABLE IV
RMSE OF VEHICLE POSITION AND CPU RUNNING TIME OF THREE ALGORITHMS

Dataset / Algorithm	In Car Park		In Victoria Park	
	RMSE (m)	Cost time (s)	RMSE (m)	Cost time (s)
UFastSLAM	0.66	40.1	12.1	181.4
TUFastSLAM	0.47	44.5	8.2	208.3
ITUFastSLAM	0.29	42.2	5.3	189.9

UFastSLAM with 30 Monte Carlo runs. The test result demonstrates that ITUFastSLAM improves the positioning accuracy of the vehicle by 38.3% and 35.4% compared with the TUFastSLAM in Car Park and Victoria Park, respectively. This result is in agreement with the comparisons in Figs. 10 and 11. Moreover, the proposed algorithm consumes less running time by 5.2% and 8.8% compared with the TUFastSLAM in Car Park and Victoria Park, respectively. Therefore, the proposed SLAM algorithm has better accuracy and computational efficiency than that of the TUFastSLAM.

VI. CONCLUSION

This paper proposed the ITUFastSLAM with the adaptive genetic resampling. This algorithm is a two-dimensional (2-D) SLAM algorithm, which can be applied to the navigation of mobile vehicles or robots, such as the unmanned aerial vehicles and autonomous underwater vehicles. To apply the algorithm to different scenarios, it is critical to establish a suitable process state and measurement functions. A kinematic model is generally selected as the process state function because many odometry sensors, such as the accelerometer and compass, are easily acquired. For the measurement function, it is generally built according to the geometrical relationship between the data (such as distance and azimuthal angle) acquired from front-end sensors (such as laser, sonar, and camera) and the state of the vehicle. It is approved that the more times the vehicle revisits the map features in unit time, the more accurate navigation result is acquired. The speed of vehicle and number of particles are two important parameters to the application of the proposed algorithm. A higher speed of the vehicle will lead to fewer re-observations in unit time, and thus reduces the estimation accuracy. To satisfy the required accuracy, more particles are needed. However, it adds the CPU processing time and thus reduces the amount of processing information in unit time. Considered this correlation, it is preferred to limit the highest vehicle speed to guarantee the required navigation accuracy. The proposed algorithm can also be applied to a 3-D scenario by augmenting the vehicle and measurement states. The main challenge of 3-D SLAM is the extra computational burdens caused by updating and registering 3-D features and updating the augmented vehicle state. It requires a higher speed processing unit and larger memory, especially in large-scale environments.

The main contributions of this paper are concluded as follows.

1) A FNE was proposed to adjust the state and measurement noises of importance sampling function (ITUKF), and it was proved to have higher accuracy than that of STF and FIS [11], [12] and a reduced computational burdens compared with that of MPAW [13].
2) An adaptive genetic resampling was proposed to substitute the conventional resampling, and it had a better estimation accuracy and computational efficiency than that with GA optimization [25] and systematic resampling [15].
3) An implementation of improved transformed unscented FastSLAM was provided, and the whole algorithm had better accuracy and computational efficiency than TUFastSLAM.

REFERENCES

[1] Z. Kurt-Yavuz and S. Yavuz, "A comparison of EKF, UKF, FastSLAM2.0, and UKF-based FastSLAM algorithms," in *Proc. IEEE 16th Int. Conf. Intell. Eng. Syst.* 2012, pp. 37–43.

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

[3] K. Murphy and S. Russell, *Rao-Blackwellised Particle Filtering for Dynamic Bayesian Networks*, New York, NY, USA: Springer, 2001, vol. 2, no. 43, pp. 499–515.

[4] 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, 2008.

[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.*, vol. 17, no. 11, pp. 3110–3120, Nov. 2016.

[6] Y. Song, Q. Li, Y. Kang, and D. Yan, "Square-root cubature FastSLAM algorithm for mobile robot simultaneous localization and mapping," in *Proc. IEEE Int. Conf. Mechatronics and Automat.*, 2012, pp. 1162–1167.

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

[8] L. Chang, B. Hu, A. Li, and F. Qin, "Transformed unscented Kalman filter," *IEEE Trans. Autom. Control*, vol. 58, no. 1, pp. 252–257, Jan. 2013.

[9] M. T. Sabet, H. M. Daniali, A. Fathi, and E. Alizadeh, "Identification of an autonomous underwater vehicle hydrodynamic model using the extended, cubature, and transformed unscented Kalman filter," *IEEE J. Ocean. Eng.*, vol. 43, no. 2, pp. 457–467, Apr. 2018.

[10] D. Jingxin and W. Qiuping, *Introduction to Modern Control Theory and Method*, Beijing, China: Tsinghua Univ. Press, 2016.

[11] Q. Ge, T. Shao, S. Chen, and C. Wen, "Carrier tracking estimation analysis by using the extended strong tracking filtering," *IEEE Trans. Ind. Electron.*, vol. 64, no. 2, pp. 1415–1424, Feb. 2017.

[12] R. Havangi, "Intelligent FastSLAM: An intelligent factorized solution to simultaneous localization and mapping," *Int. J. Humanoid Robot.* vol. 14, no. 01, 2017, Art. no. 1650026.

[13] Z. Gao, D. Mu, S. Gao, Y. Zhong, and C. Gu, "Adaptive unscented Kalman filter based on maximum posterior and random weighting," *Aerosp. Sci. Technol.*, vol. 71, pp. 12–24, 2017.

[14] A. Daniyan, G. Yu, and S. Lambotharan, "An improved resampling approach for particle filters in tracking," in *Proc. 22nd Int. Conf. Digital Signal Process.*, 2017, pp. 1–5.

[15] T. Li, M. Bolic, and P. M. Djuric, "Resampling methods for particle filtering: Classification, implementation, and strategies," *IEEE Signal Process. Mag.*, vol. 32, no. 3, pp. 70–86, May 2015.

[16] T. Lv and M. Feng, "An improved FastSLAM 2.0 algorithm based on FC&ASD-PSO," *Robotica*, vol. 35, no. 09, pp. 1795–1815, 2017.

[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] S. Mallik *et al.*, "Efficiency and cost optimized design of an induction motor using genetic algorithm," *IEEE Trans. Ind. Electron*, vol. 64, no. 12, pp. 9854–9863, Dec. 2017.

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

[20] 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.
[21] 2006. [Online]. Available: http://www-personal.acfr.usyd.edu.au/tbailey/
[22] 2006. [Online]. Available: http://www-personal.acfr.usyd.edu.au/nebot/dataset.htm
[23] M. Montemerlo and S. Thrun, *FastSLAM 2.0*, New York, NY, USA: Springer, 2007.
[24] K. Xiong, H. Y. Zhang, and C. W. Chan, "Performance evaluation of UKF-based nonlinear filtering," *Automatica*, vol. 42, no. 2, pp. 261–270, 2006.
[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] S. Ju, X. Shiqian, and P. Chengyi, *Probability and Statistics*, Beijing, China: Higher Education Press, 2008,
[27] T. Bäck, "The interaction of mutation rate, selection, and self-adaptation within a genetic algorithm," *Parallel Probl. Solving Nature*, vol. 2, pp. 87–96, 1992.

Mingwei Lin (S'18) received the B.S. degree in mechanical engineering and automation from the Fuzhou University, Fuzhou, China, in 2014. He is currently working toward the Ph.D. 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, intelligent optimization, inductive power transmission, mechatronic systems, 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, Nanjing, China, in 1991 and 1994, respectively, and the Ph.D. degree in mechanical engineering from the Zhejiang University, Hangzhou, China, in 1997.

Since 1997, he has been a Faculty with the State Key Laboratory of Fluid Power and Mechatronic Systems, Zhejiang University. 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 system, mechatronic devices, and their applications to deep-sea exploitation and exploration.

Dr. Yang was the recipient of the National Technology Invention Award (Second Prize) in 2009, the University Technology Invention Award (First Prize) of Ministry of Education of P. R. China in 2006, Science and Technology Award of Zhejiang Province (First Prize) in 2006, and Zhejiang Youth Science and Technology Award in 2009. He was selected to the Program for New Century Excellent Talents in University and New Century 151 Talent Project of Zhejiang Province (First Class). He is currently the member of 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 worked in Graz University of Technology, Graz, Austria, as Postdoctoral Research Fellow 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, underwater docking, contactless power transmission, and its applications to deep-sea exploration.