RBPF-MSIS: Toward Rao-Blackwellized Particle Filter SLAM for Autonomous Underwater Vehicle With Slow Mechanical Scanning Imaging Sonar

Ling Chen, Aolei Yang, Huosheng Hu, and Wasif Naeem

***Abstract*—Simultaneous localization and mapping (SLAM) has the potential to play a fundamental and significant role in achieving full autonomy for autonomous underwater vehicles (AUV). This article proposes a Rao-Blackwellized particle filter (RBPF) SLAM algorithm for an AUV equipped with a mechanically scanning imaging sonar (MSIS) that has a very slow scanning frequency. To tackle the issues of scan distortion and sonar data sparseness caused by the slow-sampling MSIS, the core of the algorithm is a carefully designed sliding window-based scan forming module. Then the formed scans are fed into the modified RBPF to build a consistent grid-based map thus localizing the AUV accurately. Extensive simulation and experiments are carried out to verify the proposed algorithm. The results show that the proposed algorithm outperforms existing ones in terms of the level of map consistency with the environment as well as the accuracy of pose estimation.**

***Index Terms*—Autonomous underwater vehicle, mechanical scanning image sonar, rao-blackwellized particle filter, SLAM.**

I. Introduction

Autonomous underwater vehicles (AUVs) have been applied for searching underwater resources and conducting various tasks such as underwater rescue [1], map building [2], climate change evaluation [3], and pollution monitoring [4], etc. To accomplish these tasks safely and reliably, AUVs should realize accurate localization in their operating environments, which is one of their fundamental abilities.

A. Simultaneous Localization and Mapping (SLAM)

Traditionally, doppler velocity logs (DVL) and inertial measurement unit (IMU) data are fused with acoustic long [5], short [6], and ultrashort baseline [7] or global positioning system (GPS) [8] for AUVs to localize their positions, which however require a prior knowledge of their environments. On the other hand, simultaneous localization and mapping (SLAM) algorithm enables AUVs to fuse sensor data and build a map of unknown environment while localizing them in this built map simultaneously. The perception sensors used in underwater SLAM algorithms include underwater camera, multibeam sonar, side-scan sonar, and single-beam mechanically scanning imaging sonar (MSIS).

In general, the camera-based visual SLAM algorithms conduct feature extraction and matching from the captured underwater images. Its front end calculates relative camera motion between two either adjacent images or loop-closing images and its back end conducts a bundle adjustment process to produce an optimized pose estimation [9], [10]. The advantages of using underwater camera for SLAM include low cost, ability to provide rich information besides being easily deployed and interpreted based on recent advances in image/video processing. However, the cameras can only work in clean underwater environments with good light conditions for feature extraction. Therefore, there is limited literature available on using underwater cameras in visual SLAM algorithms.

Both multibeam sonar and side-scan sonar have been widely deployed on AUVs for safe operation and underwater exploration. More specifically, a multibeam sonar emits multiple beams at different angles and obtain the range data from obstacles by analysing the returned multiple acoustic echoes. In contrast, a side scan sonar transmits the sound wave to the far side in a spherical wave after the sound pulse is emitted. After hitting the sea bottom, the reflected wave or the backscattered wave returns to the transducer along the original route. The intensity and time of the reflected wave can be detected and used to generate an image that represents the topography of the sea bottom.

The underwater images obtained by both multibeam sonar and side-scan sonar are similar to the camera images and can be used for SLAM. For instance, Barkby *et al.* [11] proposed an extended Kalman filter (EKF) SLAM algorithm by fusing motion model and the result of scan matching of multibeam sonar images. Chen *et al.* [12] implemented an EKF SLAM algorithm based on image feature points using side scan sonar. However, both multibeam and side-scan sonar sensors are bulky,

Manuscript received April 1, 2019; revised August 26, 2019; accepted August 27, 2019. This work was supported in part by the National Natural Science Foundation of China under Grants 61703262 and 61873158, and in part by the Shanghai Sailing Program under Grant 16YF1403700, and in part by the Natural Science Foundation of Shanghai under Grant 18ZR1415100. *(Corresponding author: Aolei Yang.)*

L. Chen is with the School of Engineering and Design, Hunan Normal University, Changsha 410081, China (e-mail: lcheno@hunnu.edu.cn).

A. Yang is with the School of Mechatronics Engineering and Automation, Shanghai University, Shanghai 200072, China (e-mail: aolei@shu.edu.cn).

H. Hu is with the School of Computer Science and Electronic Engineering, University of Essex, Colchester CO4 3SQ, U.K. (e-mail: hhu@essex.ac.uk).

W. Naeem is with the School of Electronics, Electrical Engineering and Computer Science, Queen's University Belfast, University Road, Belfast, BT7 1NN, Northern Ireland, U.K. (e-mail: w.naeem@qub.ac.uk).

Digital Object Identifier 10.1109/JSYST.2019.2938599

costly, and power-consuming, which limit their applications on small AUV with a limited payload.

On the other hand, MSIS has become popular in small AUVs due to its low cost, small size, low power consumption, and low computational complexity. However, it takes over 10 s to accomplish a complete scan (360°). The scanned data is noisy and sparse, making the SLAM implementation difficult. To tackle these problems, various approaches have been proposed to parsing the image data obtained from MSIS to obtain the range-angle data for SLAM. Ribas *et al.* [13] and Dong *et al.* [14] used Hough transform to extract line features by fitting the range-angle data from MSIS, and then feed them to the update phase of an EKF to realize accurate pose estimation of AUV.

B. Rao-Blackwellized Partical Filter

SLAM algorithms suffer from its enormous update complexity with an increasing number of features, a growing scale of environment and the linearization error. Therefore, Burguera proposed two similar versions of localization algorithm based on scan matching, namely uspIC [15] and μspIC [16]. A scan building module is first designed to gather a set of discretely sampled sonar reading points incorporating the vehicle motion estimated through EKF dead reckoning. Then the two consecutively built scans are registered to estimate the relative transform of these two scans, all of which are concatenated to produce the poses of AUV. However, matching consecutive scans is still likely to drift, resulting in unbounded localization errors as the vehicle travels.

Recently, Particle Filter (PF) algorithms have been widely deployed in autonomous navigation of AUVs. The most popular one is Rao-Blackwellized PF (RBPF) [17]. Its basic idea is to utilize a set of particles to estimate the posterior of robot pose. Each particle represents a potential trajectory and an individual map is associated with each particle. This novel representation allows to cope with nonlinear robot motion models of most autonomous robots. Therefore, RBPF SLAM have been successfully applied for building reliable occupancy grid maps for land-based mobile robots with laser range finders [18]. However, very few work have been conducted on utilizing RBPF to build occupancy grid map for AUVs with slow sampling MSIS.

This article presents a novel RBPF SLAM algorithm for AUV equipped with a slow sampling sonar MSIS. It is able to build an accurate occupancy grid map while providing accurate estimation of AUV poses. The occupancy grid map can be used for global localization and path planning of AUV. The main contribution of the proposed algorithm is twofold. 1) A sliding-window-based scan forming (SF) module is proposed to conquer scan distortion and sonar data sparseness caused by the slow-sampling MSIS. 2) A complete pipeline of applying RBPF SLAM is implemented to handle noisy and slow-sampled sonar readings from MSIS and produce an accurate occupancy grid map.

The rest of the article is organized as follows. Section II illustrates the problem of the slow-sampling characteristics of MSIS. The proposed RBPF SLAM algorithm for AUVs is detailed in Section III. Simulation and experiment results and analysis are given in Section IV to demonstrate the feasibility and performance of the proposed algorithm. Finally, a brief conclusion and future work are presented in Section V.

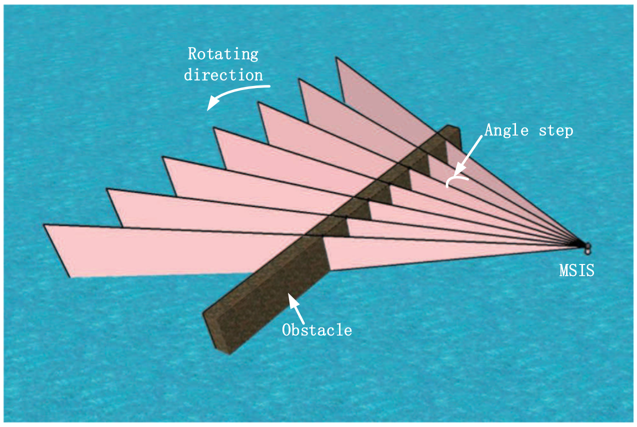


Fig. 1. MSIS sensor is rotating its sonar head for collecting range data.

II. Slow-Sampling Problem With MSIS

A MSIS sonar is installed with a transducer head driven by a mechanical motor system for rotation at preset angular steps. It conducts scans in a horizontal two-dimensional (2-D) plane during the rotation process. At each angular position, the MSIS head emits an acoustic beam which is fan-shaped with a narrow horizontal beam-width, as shown in Fig. 1. When this acoustic signal collides with any objects during its journey, it will return a certain energy whose amplitude determines the likelihood of objects. By measuring the time of flight of the returning signal, it can calculate the range using the speed of sound in water. Combining the amplitude of echo and calculated range, the sonar heads will generate a set of bins which can be visualized as an image shown in Fig. 2.

In our article, we have used a Tritech Micron sonar sensor that takes at least 6 s to accomplish one full 360° scan, which may be even longer depending on the chosen parameters such as range, angle, and step resolution of the sonar. If the AUV travels fast, the slowing scanning sonar will make the scanned image seriously distorted, as can be seen in Fig. 2. Also, its sparse data makes SLAM implementation difficult. The scan formation module based on a sliding window is proposed in this article trying to tackle these challenging issues.

III. Proposed SLAM Algorithm

Fig. 3 shows the architecture of the proposed SLAM algorithm. As can be seen, the algorithm is iterated by adopting a particle set of poses to guide the whole SLAM process. The dead reckoning module fuses data from a variety of sensors through an EKF to produce rough estimation of the AUV pose. It is used for SF process and the control input for propagating each pose particle.

The sonar reading queue Q_{SR} is generated from the preprocessed MSIS by the beam segmentation module. The dead reckoning queue Q_{DR} is used by the SF module to build a full 360°

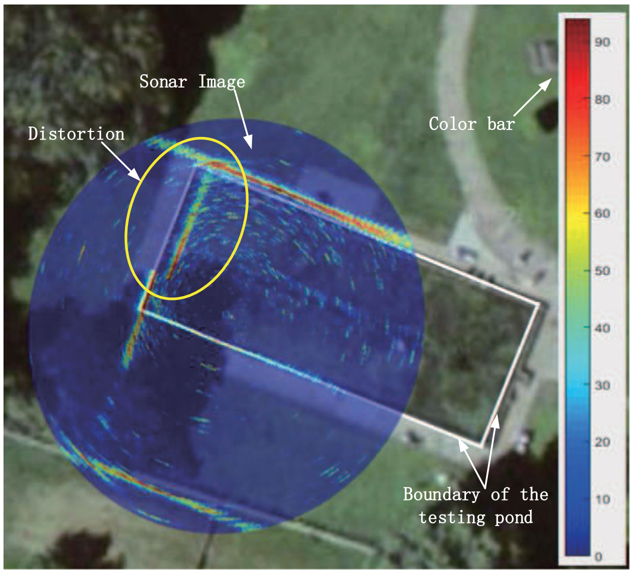


Fig. 2. 360° sonar image of a real rectangular testing pond. The color depth represents the echo amplitude of the sonar bins. As the AUV is moving while the image is generating, a distortion of the scan appears.

scan of segmented sonar readings. The reason for generating a full 360° of sonar readings lies with the fact that the subsequent calculation of scan matching is more accurate and robust when a big angle range of scanned data is incorporated. The formed scan is then used for motion update and scan matching in the RBPF SLAM process.

Incorporating dead reckoning into the SF module is beneficial for getting rid of the sonar scan distortion resulted from the significant AUV motion and the slow sampling MSIS. Following the pipeline of sampling from dead reckoning, importance weighting, resampling, and map estimation, the core part of the particle filter-based SLAM module utilizes the formed scans z_t and dead reckoning u_t to estimate both AUV poses and a grid map of the environment iteratively. The detailed description of each module in the architecture is presented in the following subsections.

A. Dead Reckoning

Dead reckoning module aims to provide the rough estimation of the AUV pose using various sensors namely inertial measurement unit, altimeter, and DVL if available. The dead reckoning module applies an EKF for pose estimation by fusing data from those sensors while the sonar is sampling data.

1) Process Prediction: Let the vehicle state $\boldsymbol{X}_k = [\boldsymbol{p}_k^T, \boldsymbol{\varphi}_k^T]^T$ in the global coordinate frame be estimated at time k, where $\boldsymbol{p}_k$ represents the vehicle position, and $\boldsymbol{\varphi}_k$ represents the vehicle attitude of AUV. Specifically, $\boldsymbol{p}_k$ and $\boldsymbol{\varphi}_k$ are defined as

$$\boldsymbol{p}_k = \begin{bmatrix} x_k & y_k & z_k \end{bmatrix}^T, \boldsymbol{\varphi}_k = \begin{bmatrix} \phi_k & \theta_k & \psi_k \end{bmatrix}^T$$

where x_k, y_k, z_k are the position coordinates in each axis in the global coordinate frame and ϕ_k, θ_k, ψ_k are Euler angles roll, pitch, and yaw in each corresponding axis.

Let the linear and angular velocities of AUV be $\boldsymbol{v}_k$ and $\boldsymbol{\omega}_k$, which are grouped to form as the control input $\boldsymbol{u}_k = [\boldsymbol{v}_k^T, \boldsymbol{\omega}_k^T]^T$. Specifically, $\boldsymbol{v}_k$ and $\boldsymbol{\omega}_k$ are represented as

$$\boldsymbol{v}_k = \begin{bmatrix} u_k & \nu_k & w_k \end{bmatrix}^T \quad \boldsymbol{\omega}_k = \begin{bmatrix} o_k & q_k & r_k \end{bmatrix}^T$$

where each element in the two vectors is the linear and angular velocity in each axis of the body frame of the AUV. Then, the process model for AUV can be expressed as a nonlinear discrete time system

$$\boldsymbol{X}_{k+1} = f(\boldsymbol{X}_k, \boldsymbol{u}_k) = \boldsymbol{X}_k + \Delta_T J(\boldsymbol{X}_k)\boldsymbol{u}_k \tag{1}$$

where

$$J(\boldsymbol{X}) = \begin{bmatrix} c\psi c\theta & c\psi s\theta s\phi - s\psi c\phi & c\psi s\theta c\phi + s\psi s\phi & 0 & 0 & 0 \\ s\psi c\theta & s\phi s\psi s\theta + c\psi c\phi & s\psi s\theta c\phi + s\phi c\psi & 0 & 0 & 0 \\ -s\theta & c\theta s\phi & c\theta c\phi & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & s\phi t\theta & c\phi t\theta \\ 0 & 0 & 0 & 0 & c\phi & -s\phi \\ 0 & 0 & 0 & 0 & s\phi / c\theta & c\phi / c\theta \end{bmatrix}$$

is the transformation matrix, and Δ_T is the sampling time interval. Note that, for simplicity, s, c, and t in the above matrix are $\sin$, $\cos$, and $\tan$ functions, respectively.

If DVL is available, $\boldsymbol{u}_k$ can be represented by the measurement from DVL corrupted with an additive Gaussian noise $\boldsymbol{w}_k \sim \mathcal{N}(\mathbf{0}, \boldsymbol{Q}_{\boldsymbol{u}})$. Whereas in the absence of DVL, $\boldsymbol{u}_k$ is assumed to be a Gaussian noise $\boldsymbol{w}_k \sim \mathcal{N}(\boldsymbol{C}, \boldsymbol{Q}_{\boldsymbol{u}})$ with a constant mean $\boldsymbol{C}$. As a result of this noise, the error of the dead reckoning will be accumulated and further unbounded. Thus, other sensor information is called in the update phase of EKF for correcting the error.

By virtue of (1), the vehicle state is estimated below

$$\hat{\boldsymbol{X}}_{k+1|k} = f(\hat{\boldsymbol{X}}_{k|k}, \hat{\boldsymbol{u}}_k) \tag{2}$$

and the covariance matrix of the prediction error is provided by

$$\mathbf{P}_{k+1|k} = \mathbf{F}_{k+1}\mathbf{P}_{k|k}\mathbf{F}_{k+1}^{\mathrm{T}} + \mathbf{G}_{k+1}\boldsymbol{Q}_{\boldsymbol{u}}\mathbf{G}_{k+1}^{\mathrm{T}} \tag{3}$$

where $\mathbf{F}_{k+1}$ and $\mathbf{G}_{k+1}$ are the Jacobian matrices formed by taking the partial derivatives of the nonlinear model function f with respect to the state $\hat{\boldsymbol{x}}_k$ and the noise $\boldsymbol{w}_k$, respectively.

2) Update With Altimeter: The depth of the AUV in underwater can be used for providing the z_k measurement in $\boldsymbol{p}_k$, and the depth measurement equation can be expressed as

$$z_{d,k} = \mathbf{H}_{d,k}\boldsymbol{X}_k + \sigma_{d,k} = \begin{bmatrix} 0 & 0 & 1 & \mathbf{0}_{1\times 3} \end{bmatrix} \boldsymbol{X}_k + \sigma_{d,k} \tag{4}$$

where $\sigma_{d,k}$ is a zero-mean Gaussian noise with the covariance being R_d.

3) Update With Orientation From IMU: The IMU is able to provide the orientation measurement whose model can be expressed as

$$\boldsymbol{z}_{a,k} = \mathbf{H}_{a,k}\boldsymbol{X}_k + \boldsymbol{\sigma}_{a,k} = \begin{bmatrix} \mathbf{0}_{3\times 3} & \boldsymbol{I}_{3\times 3} \end{bmatrix} \boldsymbol{X}_k + \boldsymbol{\sigma}_{a,k} \tag{5}$$

where $\boldsymbol{I}$ represents the 3×3 identity matrix and $\boldsymbol{\sigma}_{a,k}$ is a zero-mean Gaussian noise with the covariance being $\boldsymbol{R}_a$. Then the model prediction is updated by applying standard EKF update equations to produce the dead reckoning estimation.

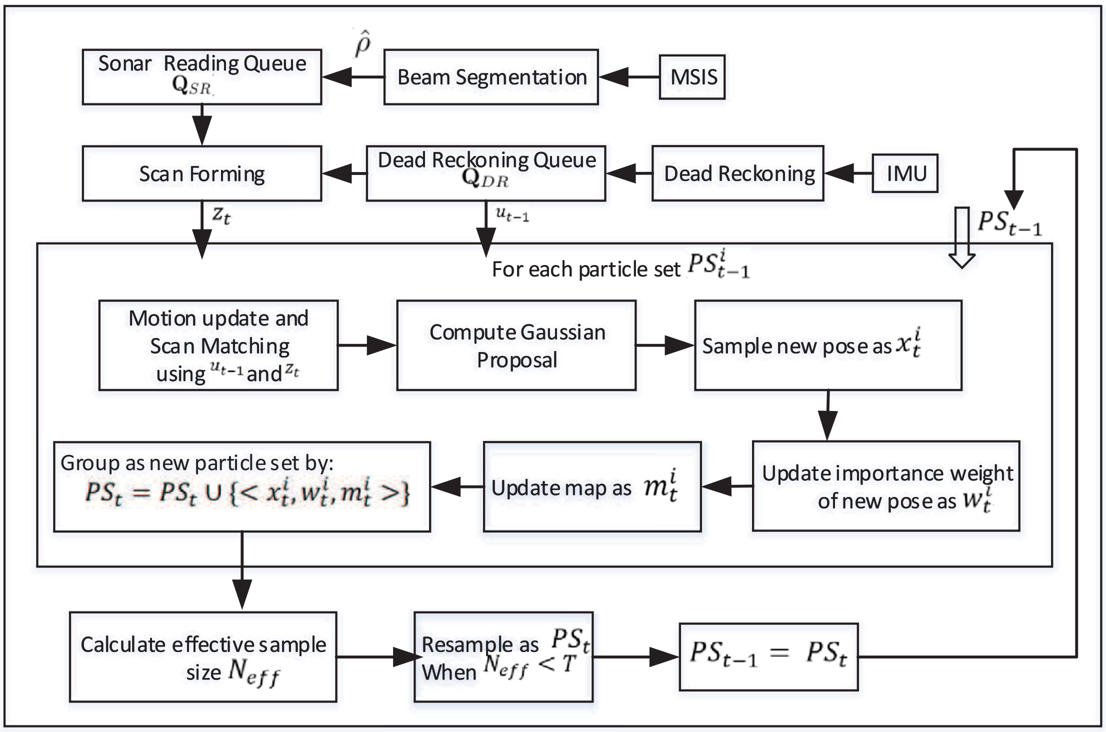

Sonar Reading Queue $\mathbf{Q}_{SR}$
$\hat{\rho}$
Beam Segmentation
MSIS
Scan Forming
Dead Reckoning Queue $\mathbf{Q}_{DR}$
Dead Reckoning
IMU
z_t
u_{t-1}
PS_{t-1}
For each particle set PS^i_{t-1}
Motion update and Scan Matching using u_{t-1} and z_t
Compute Gaussian Proposal
Sample new pose as x^i_t
Group as new particle set by:
$PS_t = PS_t \cup \{< x^i_t, w^i_t, m^i_t >\}$
Update map as m^i_t
Update importance weight of new pose as w^i_t
Calculate effective sample size N_{eff}
Resample as PS_t When $N_{eff} < T$
$PS_{t-1} = PS_t$

Fig. 3. Architecture of the proposed algorithm.

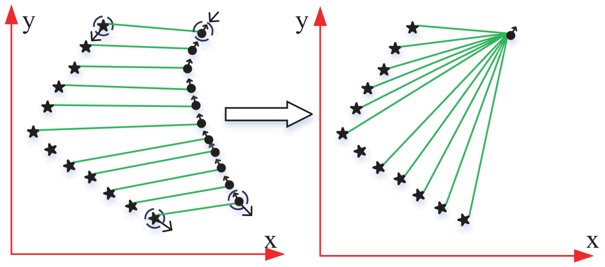


Fig. 4. Diagram of the SF process based on the sliding window strategy.

B. *Scan Forming Based on a Sliding Window*

The SF module can apply the results of dead reckoning to form a full 360° scan as if all scanned points are read instantaneously. A sliding window strategy is employed to govern the SF. Two queues, namely Q_{SR} and Q_{DR}, are used to maintain both the history of dead reckoning and sonar reading. The size of the queue is equal to the number of the scanned sonar points of a full 360° scan. When a new sonar point is obtained, it is pushed back into the rear of Q_{SR} while the front one is pushed out. Simultaneously, the dead reckoning pose corresponding to the new sonar point is also pushed back into the rear of Q_{DR} while the front one is pushed out.

This strategy deploys historical sonar readings to solve data deficiency caused by the slow sampling MSIS. Fig. 4 shows the SF process that utilizes a sliding window strategy, in which the black stars represent the preprocessed sonar readings and the black dots represent the dead reckoning poses.

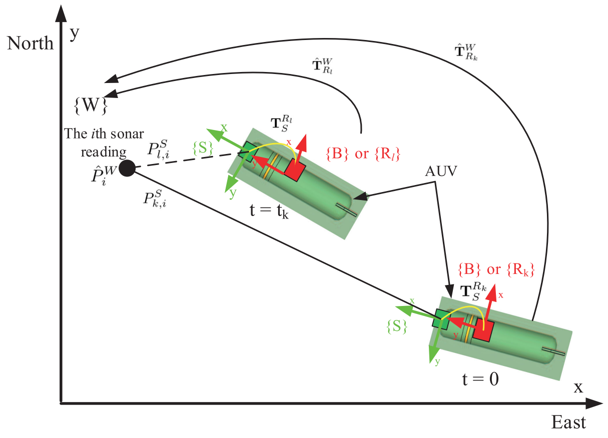


Fig. 5. Representation of different coordinate systems during SF.

Each sonar point $P^S_{k,i}$ in the queue of Q_{SR} is transformed from the current robot body frame $\{R_k\}$ to the previous robot body frame, namely $\{R_l\}$. This means that all the scanned points will be represented in frame $\{R_l\}$, as shown in Fig. 5. The transformation is based on the fact that the sonar reading is kept the same in the world coordinate frame $\{W\}$, which can be described by the following equation:

$$\hat{P}^W_i = \hat{\mathbf{T}}^W_{R_l} \oplus \mathbf{T}^{R_l}_S \oplus P^S_{l,i} = \hat{\mathbf{T}}^W_{R_k} \oplus \mathbf{T}^{R_k}_S \oplus P^S_{k,i} \quad (6)$$

where $\hat{P}^W_i$ is the ith sonar reading with respect to the world frame $\{W\}$, $\hat{\mathbf{T}}^W_{R_l}$ is the last transform of the robot body frame $\{R_l\}$ with respect to the world frame $\{W\}$ and the similar

Algorithm 1: Scan Forming.

```
1  $[S_{\text{cur}}, Q_{DR}, Q_{SR}] = SF(\hat{\rho}, \mathbf{P}_\rho, X_k, Q_{DR}, Q_{SR})$ {
2  $Q_{DR} = Q_{DR}.Pop()$;
3  $Q_{DR}.PushBack(X_k)$;
4  $Q_{SR} = Q_{SR}.Pop()$;
5  $Q_{SR}.PushBack(\hat{\rho})$;
6  $S_{cur} = [\ ]$;
7  $i = 0$;
8  $\hat{\mathbf{T}}^W_{R_l} = Q_{DR}.Back()$;
9  for All elements in $Q_{DR}, Q_{SR}$ do
10     $\hat{\mathbf{T}}^W_{R_k} = Q_{DR}(i)$;
11     $\hat{\rho}_i = Q_{SR}(i)$;
12     $\hat{P}^S_{l,i} = \ominus\mathbf{T}^{R_l}_S \ominus \hat{\mathbf{T}}^W_{R_l} \oplus \hat{\mathbf{T}}^W_{R_k} \oplus \mathbf{T}^{R_k}_S \oplus P2C(\hat{\rho}_i)$;
13     $\mathbf{P}_{R_l} = \mathbf{J}_{l1\oplus}\mathbf{J}_{l\ominus}\mathbf{P}_{Wl}\mathbf{J}^{\mathrm{T}}_{l\ominus}\mathbf{J}^{\mathrm{T}}_{l1\oplus} + \mathbf{J}_{l2\oplus}\mathbf{P}_W\mathbf{J}^{\mathrm{T}}_{l2\oplus}$;
14     $S_{\text{cur}} = S_{\text{cur}} \cup \{[\hat{P}^S_{l,i}, \mathbf{P}_{R_l}]\}$;
15     $i = i + 1$;
16 return $S_{\text{cur}}, Q_{DR}, Q_{SR}$;
17 }
```

representation goes to $\mathbf{T}^{R_l}_S, \hat{\mathbf{T}}^W_{R_k}, \mathbf{T}^{R_k}_S$ with $\{S\}$ being the sensor frame and $\{R_k\}$ being the kth robot body frame.

As the transform from the sensor frame to the robot body frame is static, we have $\mathbf{T}^{R_l}_S = \mathbf{T}^{R_k}_S$. From (6), the point $P^S_{k,i}$ can be represented with respect to the body frame of the last sonar $\{R_l\}$ as

$$\hat{P}^S_{l,i} = \ominus\mathbf{T}^{R_l}_S \ominus \hat{\mathbf{T}}^W_{R_l} \oplus \hat{\mathbf{T}}^W_{R_k} \oplus \mathbf{T}^{R_k}_S \oplus P^S_{k,i} \tag{7}$$

where $\oplus$ and $\ominus$ are the compounding and inversion transformations proposed in [19].

Algorithm 1 describes the SF process based on the sliding window. The input parameters are the new sonar reading $\hat{\rho}$, covariance of the sonar reading $\mathbf{P}_\rho$, current dead reckoning $\boldsymbol{X}_k$, last dead reckoning $\boldsymbol{X}_{k-1}$, current estimated robot pose $\hat{\boldsymbol{X}}_k$, the queue of dead reckoning Q_{DR} and the queue of sonar reading Q_{SR}. The formed scan is denoted as S_{cur} which is also the returned value of the algorithm. $\mathbf{P}_W$ is the covariance of current sonar reading represented in global frame $\{W\}$, $\mathbf{J}_{l1\oplus}$, $\mathbf{J}_{l\ominus}$, and $\mathbf{J}_{l2\oplus}$ are the Jacobian matrices of $\hat{P}^S_{l,i}$ with respect to $\hat{\mathbf{T}}^W_{R_k}$ and $\hat{\mathbf{p}}^{\mathbf{W}}_{\mathbf{i}}$.

C. RBPF SLAM With MSIS

The whole flowchart of the proposed algorithm is presented in Algorithm 2. A is the size of the queue for both Q_{SR} and Q_{DR}. After a full scan S_{cur} has been formed by abovementioned Algorithm 1, it is then considered as a measurement z_t and fed into a RBPF SLAM pipeline. The SLAM problem is solved by factorizing the SLAM process into separate localization and mapping parts, which stems from the theory that a joint probability can be converted to the product of conditional probability

$$p(x_{1:t}, m|z_{1:t}, u_{1:t-1}) = p(m|x_{1:t}, z_{1:t}) \cdot p(x_{1:t}|z_{1:t}, u_{1:t-1}) \tag{8}$$

where $p(x_{1:t}|z_{1:t}, u_{1:t-1})$ is the posterior of about potential trajectories $x_{1:t}$ of the robot given its observations $z_{1:t}$ and its odometry measurements $u_{1:t}$, $p(m|x_{1:t}, z_{1:t})$ is the posterior over maps, and $p(x_{1:t}, m|z_{1:t}, u_{1:t-1})$ is the posterior over maps and trajectories. As $p(m|x_{1:t}, z_{1:t})$ can be computed analytically [15] given the knowledge of $x_{1:t}$ and $z_{1:t}$, the key is to compute $p(x_{1:t}|z_{1:t}, u_{1:t-1})$.

To estimate the posterior $p(x_{1:t}|z_{1:t}, u_{1:t-1})$, a set of particles are utilized. Each particle is composed of robot pose x, weight w, and the grid map m. Then the particle filter algorithm incrementally utilizes the available dead reckoning and sonar scan data to update the set of particles which represents the posterior over the map and the trajectory of the AUV. The process includes four steps, *sampling*, *importance weighting*, *resampling*, and *map updating* [18]. The function $\hat{x}^{(i)}_t = SM(m^{(i)}_{t-1}, z_t, x'^{(i)}_{t-1})$ stands for scan matching and is used for finding the most likely pose that matches against the current map $m^{(i)}_{t-1}$ with current observation z_t and $x'^{(i)}_{t-1}$ being the initial estimation [20].

Algorithm 2: RBPF_MSIS.

```
1  $Q_{DR} = \Phi$
2  $Q_{SR} = \Phi$;
3  $\mathbf{PS}_{t-1} = Init()$;
4  for each $(\hat{\rho}, \mathbf{P}_\rho) = SonarReading()$ do
5      $X_k = DeadReckoning()$;
6      $a = Q_{SR}.Size()$;
7      if $a < A$ then
8          $Q_{DR}.PushBack(\mathbf{X}_k)$;
9          $Q_{SR}.PushBack(\hat{\rho})$;
10         Continue;
11     $[S_{\text{cur}}, Q_{DR}, Q_{SR}] = SF(\hat{\rho}, \mathbf{P}_\rho, X_k, Q_{SR}, Q_{DR})$;
12     $z_t = S_{\text{cur}}$;
13     $u_{t-1} = X_k \ominus X_{k-1}$;
14     for all $\mathbf{PS}^{(i)}_{t-1} \in \mathbf{PS}_{t-1}$ do
15         $< x^{(i)}_{t-1}, w^{(i)}_{t-1}, m^{(i)}_{t-1} >= \mathbf{PS}^{(i)}_{t-1}$;
16         //sampling
17         $x'^{(i)}_t = x^{(i)}_{t-1} \oplus u_{t-1}$; //motion model
18         //importance weighting
19         $\hat{x}^{(i)}_t = SM(m^{(i)}_{t-1}, z_t, x'^{(i)}_{t-1})$; //scan matching
20         if $\hat{x}^{(i)}_t = \phi$ then
21             $x^{(i)}_t = x'^{(i)}_t$;
22             $w^{(i)}_t = w^{(i)}_{t-1} p(z_t|m^{(i)}_{t-1}, x^i_t)$;
23         else
24             //resampling
25             $[x^{(i)}_t, w^{(i)}_t] = GPC(\hat{x}^{(i)}_t, \Omega, u_{t-1}, m^{(i)}_{t-1})$;
26         //map updating
27         $m^{(i)}_t = UpdateMap(m^{(i)}_{t-1}, z_t, x^{(i)}_t)$;
28         $\mathbf{PS}_t = \mathbf{PS}_t \cup \{x^{(i)}_t, w^{(i)}_t, m^{(i)}_t\}$;
29     $N_{eff} = \frac{1}{\sum_{i=1}^{N}(\tilde{w}(i)^2)}$
30     if $N_{eff} < T$ then
31         $\mathbf{PS}_t = resample(\mathbf{PS}_t)$
32     $\mathbf{PS}_{t-1} = \mathbf{PS}_t$;
```

Algorithm 3: Gaussian Proposal Calculation.

```
1  $[x_t^{(i)}, w_t^{(i)}] = GPC(\hat{x}_t^{(i)}, \Omega, u_{t-1}, m_{t-1}^{(i)})\{$
2  for $k = 1, ..., K$ do
3      $x_k \sim \{x_j |\ |x_i - \hat{x}^{(i)}| < \Omega\}$;
4  $\mu_t^{(i)} = (0, 0, 0)^T$;
5  $\eta^{(i)} = 0$;
6  for all $x_j \in \{x_1, ..., x_K\}$ do
7      $\mu_t^{(i)} = \mu_t^{(i)} + x_j \cdot p(z_t|m_{t-1}^{(i)}, x_j) \cdot p(x_t|x_{t-1}^{(i)}, u_{t-1})$;
8      $\eta^{(i)} = \eta^{(i)} + p(z_t|m_{t-1}^{(i)}, x_j) \cdot p(x_t|x_{t-1}^{(i)}, u_{t-1})$;
9  $\mu_t^{(i)} = \mu_t^{(i)} / \eta^{(i)}$;
10 $\Sigma_t^{(i)} = 0$;
11 for all $x_j \in \{x1, ..., x_K\}$ do
12     $\Sigma_t^{(i)} = \Sigma_t^{(i)} + (x_j - \mu^{(i)})(x_j - \mu^{(i)})^T \cdot$
       $p(z_t|m_{t-1}^{(i)}, x_j) \cdot p(x_j|x_{t-1}^{(i)}, u_{t-1})$;
13 $\Sigma_t^{(i)} = \Sigma_t^{(i)} / \eta^{(i)}$;
14 $x_t^{(i)} \sim N(\mu_t^{(i)}, \Sigma_t^{(i)})$;
15 $w_t^{(i)} = w_{t-1}^{(i)} \cdot \eta^{(i)}$;
16 }
```

Algorithm 3 shows the Gaussian proposal calculation pipeline which aims to calculate the Gaussian approximation of the proposal distribution, based on which the new particle is sampled for next iteration. Ω is the interval threshold for selecting the points around the neighbouring area of $\hat{x}_t^{(i)}$. The likelihood of an observation based on the map and pose $p(z_t|m_{t-1}^{(i)}, x_j)$ is computed by using a "beam endpoint model" proposed by [21] instead of "full beam model." The reason is twofold, as follows: 1) the computational complexity of "full beam model" is much higher than that of "beam endpoint model"; 2) as there are not many dynamic obstacles in our testing scenarios, the "beam endpoint model" becomes unnecessary. The term $p(x_j|x_{t-1}^{(i)}, u_{t-1})$ can be calculated using the odometry motion model proposed in [22]. The map $m_t^{(i)}$ is then updated with the drawn pose $x_t^{(i)}$ and the observation z_t. The map and pose of the particle with the biggest weight are then chosen as the built map and the estimated AUV pose.

IV. Experimental Results

Both simulation and practical experiments are conducted with two different AUVs and testing sites to verify the effectiveness of the proposed RBPF-MSIS algorithm. Table I shows the main parameters of our algorithm for both simulation and practical experiments.

A. Simulation

1) Simulation Setup: Fig. 6 shows the simulation setup for our AUV. As can be seen, our AUV is equipped with various sensors such as IMU, DVL (frame is denoted as $\{I\}$), and a forward looking MSIS (frame is denoted as $\{S\}$). The AUV body frame is represented with $\{B\}$ or $\{R\}$ while the global frame is denoted as $\{W\}$ whose axis x points to East, axis y to North and axis z to the up direction.

TABLE I
The Parameters of the RBPF-MSIS Algorithm

Parameter	Value
Map resolution	0.25 m
Number of particles	120
Resample threshold, T	0.5
Interval threshold for selecting points around the estimated pose $\hat{x}_t^{(i)}$, Ω	1.5 m
The number of iterations of the scan matcher	5

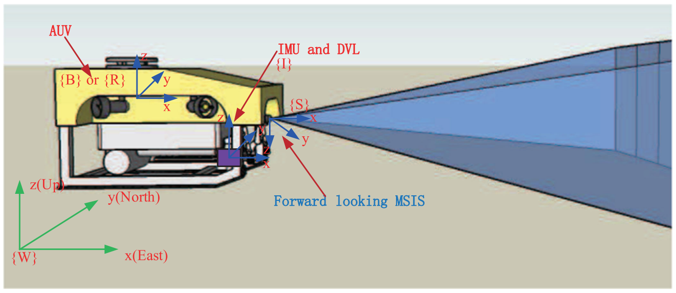


Fig. 6. Simulation setup for our AUV.

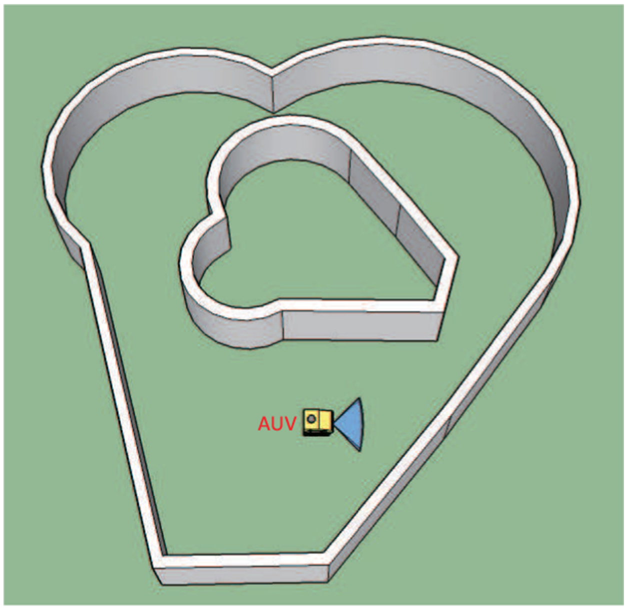


Fig. 7. Simulated environment for AUV manoeuvring.

Fig. 7 shows the simulation environment which is an anomalous-shaped pond where the AUV is controlled to navigate around. The 3-D model of the environment is loaded into the 3-D simulator *Gazebo* [23]. Robot operating system (ROS) [24] is used in our simulation for convenience in interfacing with *Gazebo*. Drivers for IMU, DVL, and MSIS are loaded as plug-ins of *Gazebo* for publishing ROS-compatible data which is subscribed by the proposed SLAM algorithm. During the simulation, our AUV is controlled to navigate around the middle object for one round. The speed of the AUV is set as 0.1 m/s which is added by a Gaussian noise whose standard variance is

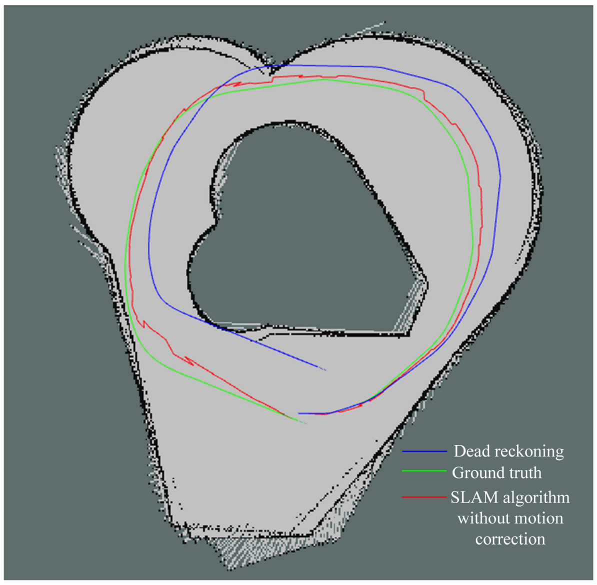


Fig. 8. Occupancy grid map and trajectories produced by SLAM algorithm with SF without incorporating motions.

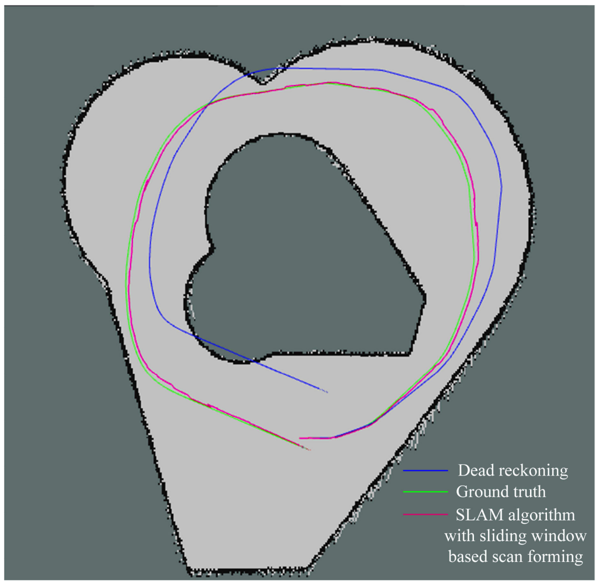


Fig. 10. Occupancy grid map and trajectories produced by SLAM algorithm with sliding-window-based SF incorporating motions.

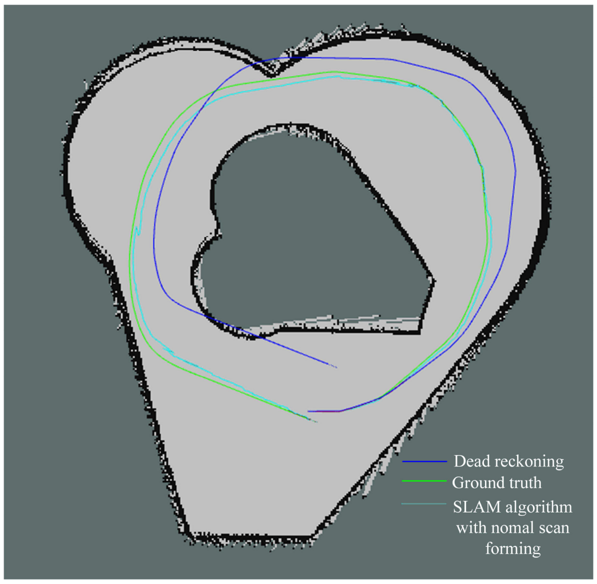


Fig. 9. Occupancy grid map and trajectories produced by SLAM algorithm with normal SF incorporating motions.

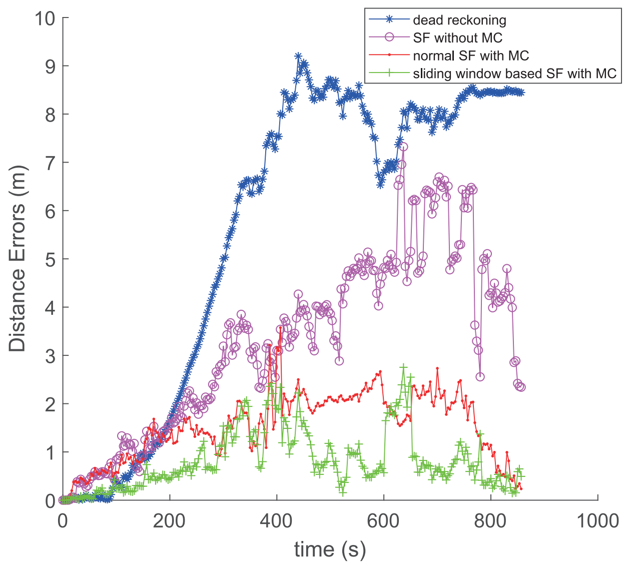


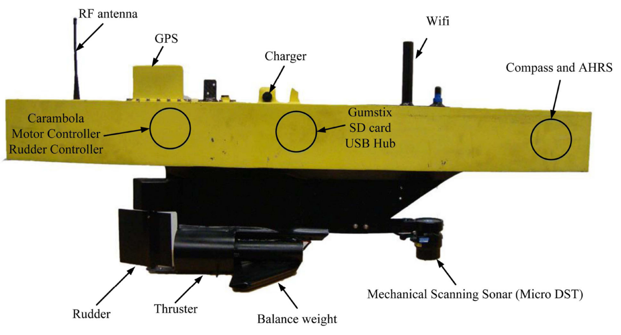
Fig. 11. Relative localization errors between the poses estimated by various algorithms and the ground truth.

0.02 m/s. The outputs of the algorithm include an occupancy grid map and an estimated AUV trajectory.

2) Simulation Result: To verify the effectiveness of the proposed algorithm, we implemented different SLAM algorithms in simulation, e.g., no SF incorporating vehicle motion, normal SF, and sliding-window-based SF. Figs. 8–10 show the occupancy grid maps and trajectories produced by these algorithms. The green line indicates the ground truth trajectory while the blue line represents the dead reckoning trajectory. The trajectories generated by each of the algorithm are represented with different colors, with the red one being SLAM trajectory without SF incorporating motion, the cyan one being the SLAM trajectory with normal SF, and the purple one being the SLAM trajectory with sliding-window-based SF.

In contrast to sliding-window-based SF, normal SF is the process of forming a full scan after the whole set of 360° sonar readings have been collected. It can be seen from the Fig. 8 that the occupancy grid map produced by the SLAM algorithm which does not use SF incorporating motion exhibits serious ghost shadow and is inconsistent with the environment.

Thanks to the SF process which incorporates vehicle motion, Fig. 9 shows the occupancy grid map generated by the SLAM algorithm with normal SF, which has much smaller ghost effect than Fig. 8. Since the sliding-window-based SF takes the



(a)

(b)

Fig. 12. Testing site for our homemade underwater vehicle which is gathering data. (a) AUV platform: SCPBot. (b) Vehicle gathering data in a rectangular pond.

advantages of the history sonar data sufficiently, the occupancy grid map built by the SLAM algorithm with sliding-window-based SF has the smallest ghost shadow and matches best with the environment. As shown in Fig. 10, the trajectory produced by the SLAM with sliding-window-based SF is closest to the ground truth, compared with trajectories given by the SLAM algorithms with normal SF and without SF.

Fig. 11 shows the deviation of trajectories by different algorithms from the ground truth in detail. The deviation is represented by the Euclidean relative localization error between the trajectories produced by abovementioned different SLAM algorithms and the ground truth. It is clear that the estimation error of dead reckoning (blue star line) is the largest due to accumulated integration error caused by sensor noises. The distance error of SLAM with SF without incorporating motion (magenta circle line) is smaller than dead reckoning, but it is still far from ideal.

The distance error of SLAM with normal SF and motion correction (MC) is smaller than the errors of the SLAM without SF and dead reckoning. In contrast, the SLAM with sliding-window-based SF with MC (red point line) has the smallest distance error. The root mean squared error (RMSE) proposed in [25] for dead reckoning, SLAM with SF without MC, SLAM with normal SF with MC and SLAM with sliding-window-based SF with MC are evaluated to be 6.47, 3.74, 1.67, and 1.02 m, respectively. Therefore, both the distance error and translational RMSE indicate that the proposed SLAM algorithm with sliding-window-based SF with MC produces the most accurate pose estimation.

B. Experiments With a Homemade AUV

To further examine the proposed SLAM algorithm, we conducted practical experiments using a homemade AUV, which is named as SCPBot and shown in Fig. 12(a). An embedded system is deployed on our AUV to collect sensor data from IMU and MSIS. The MSIS is a product named as Micron DST from the Tritech International Limited. IMU is used for providing three-dimensional (3-D) orientation measurement. The MSIS is controlled to rotate continuously in the clockwise scanning direction (from 0 to 360°) to sample sonar data. The sampling range of MSIS is set as 20 m and the rotation step angle is configured as 1.8°.

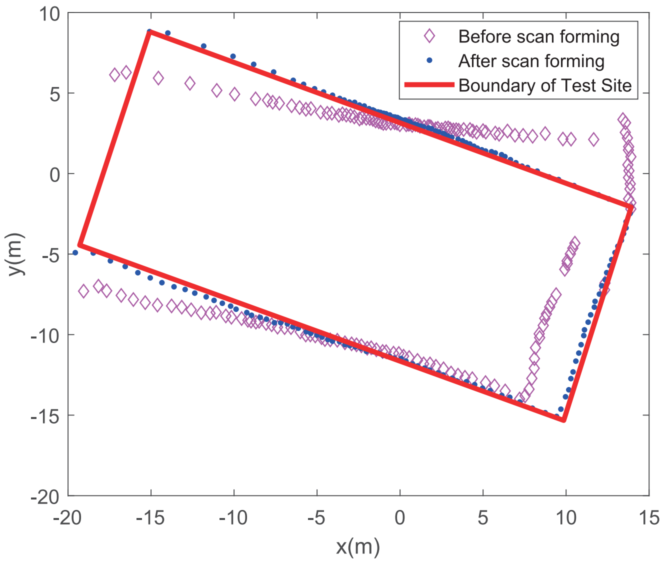
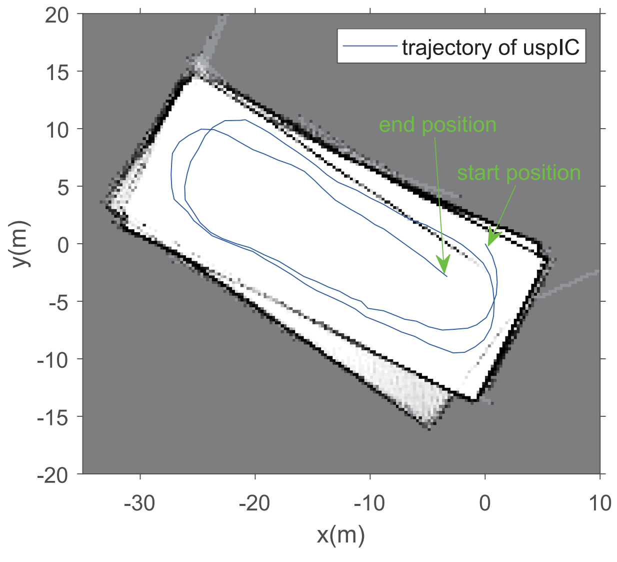


Fig. 13. Effect of SF.

We use a joystick to send control signals to our AUV via its wireless antenna above the water surface so that the vehicle can be manoeuvred remotely for data collection. The testing site is a rectangular pond shown in Fig. 12(b) where the vehicle is teleoperated to swim for two cycles. As the GPS signal is not accurate enough, the starting position and the stopping position is deliberately controlled to be the same in order to verify the accuracy of localization. Preprocessed by using dynamic threshold-based beam segmentation algorithm proposed by [13], the sonar readings together with IMU data are imported into the proposed RBPF-SLAM algorithm pipeline.

Fig. 13 presents the effect of SF, in which the sonar scans (blue dot) coincides with the boundary of the testing site. It is clear that the SF strategy can effectively eliminate the distortion (pink diamond) caused by the slow-sampling MSIS.

Fig. 14 shows the the occupancy grid map rendered with the poses estimated by uspIC [15] and the map built by the proposed



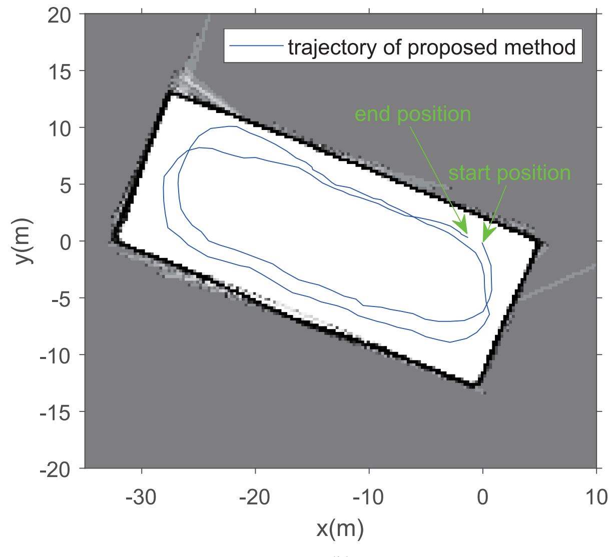
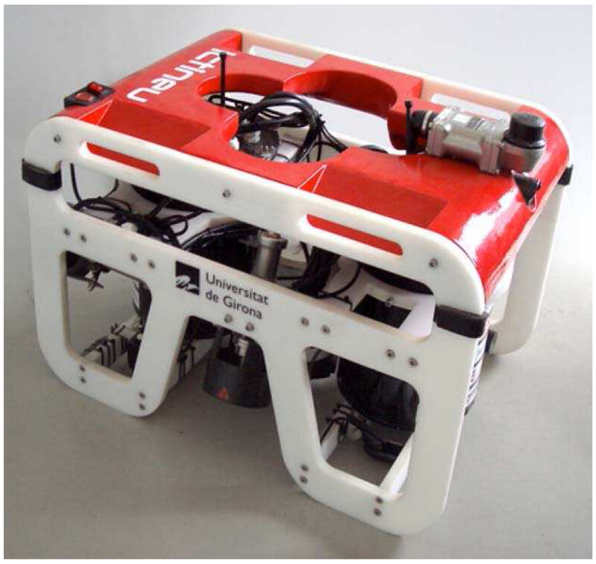


Fig. 14. Occupancy grid maps rendered with poses estimated by uspIC and built by the proposed algorithm. (a) Occupancy grid map rendered with poses estimated by uspIC. (b) Occupancy grid map built by the proposed algorithm.


(a)

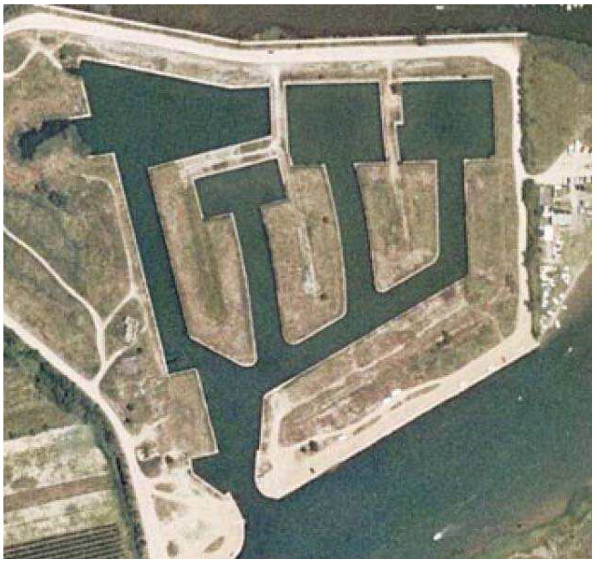

(b)

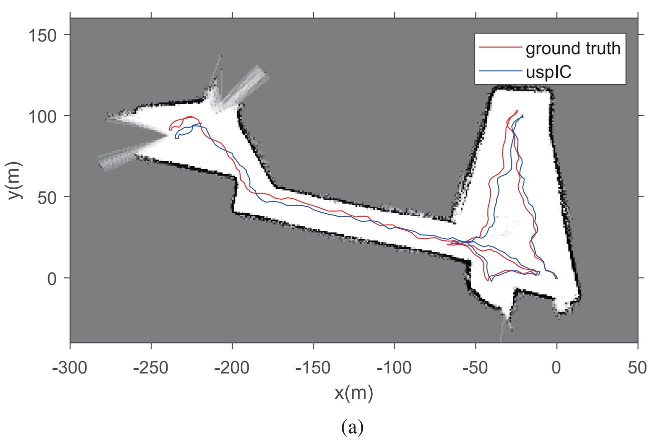
Fig. 15. Ictineu AUV and the place where data were collected [13]. (a) Ictineu AUV developed in the University of Girona. (b) Place where data were collected.

algorithm. It can be seen that the occupancy grid map rendered by the uspIC is seriously distorted and inconsistent with the rectangular pond wall due to the pose estimation error. The occupancy grid map built by the proposed SLAM algorithm matches well with the real rectangular pond wall. Although DGPS can be used to obtain absolute ground truth position of the AUV, it was not available for our testing as the base station is far away from the receiver on our AUV.

Therefore, we used the distance error between the start position and end position of the trajectory to evaluate the accuracy of different algorithms. The relative localization error between the start position and the end position of the trajectory estimated by dead reckoning, uspIC, and the proposed RBPF-SLAM algorithm were 7.77, 4.37, and 1.29 m, respectively. This further verified that the proposed RBPF-SLAM algorithm provides the most accurate pose estimation.

C. Experiment Using a Real Dataset in [13]

To extensively validate the effectiveness of the proposed algorithm, external data obtained by Ribas [13] is also used for validation. Ictineu AUV developed in University of Girona, as



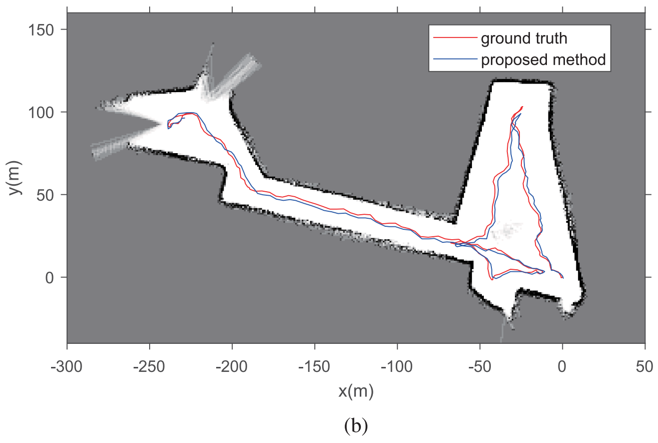


Fig. 16. Occupance grid maps rendered with the poses estimated by uspIC and the proposed algorithm using dataset in [13]. (a) Occupancy grid map rendered with poses estimated by uspIC. (b) Occupancy grid map built by the proposed algorithm.

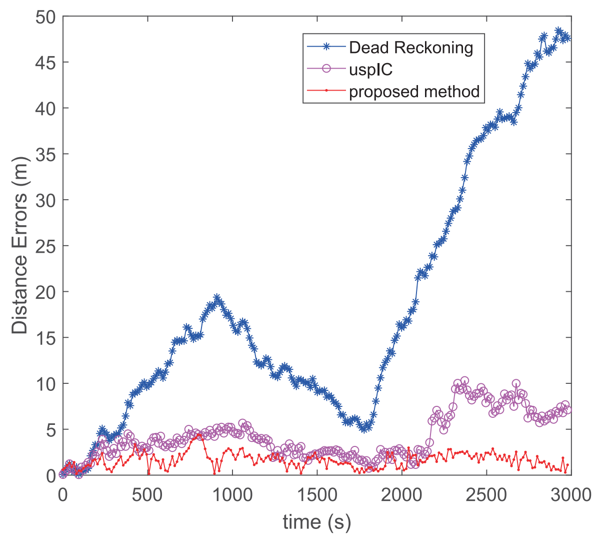


Fig. 17. Relative localization errors among the uspIC trajectories, the proposed SLAM algorithm and the ground truth using data from [13].

TABLE II
COMPUTATION TIME OF THE RBPF-MSIS ALGORITHM FOR EXPERIMENTAL PLATFORMS

	SCPBot	Ictineu AUV
Time to obtain dataset	17 min 10 s	55 min 44 s
Time to run RBPF-MSIS	4 min 20 s	14 min 6 s
Time to run uspIC	6 min 40 s	20 min 10 s

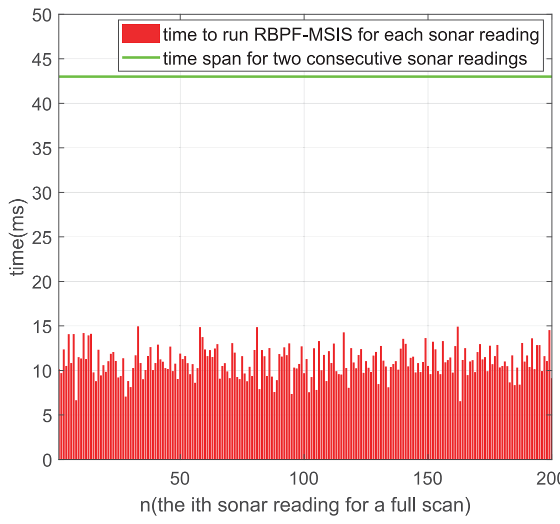


Fig. 18. Time for running RBPF-MSIS after each sonar reading has been received during a full 360° scan with SCPBot AUV.

shown in Fig. 15(a), was used to gather data in an abandoned marina located near St. Pere Pescador in the Costa Brava, Spain [see Fig. 15(b)]. The AUV was teleoperated to maneuver at an average speed of 0.2 m/s and the whole trajectory is 600 m. The experiment lasted 55 min 44 s.

The trajectory comprises a small loop and a long straight path with 200 m. The dataset contained measurements from a DVL, a compass, a motion reference unit (MRU) and MSIS. The MSIS was configured as a sampling range of 50 m, a resolution of 0.1 m, and step angle of 1.8°. In addition, a buoy with a DGPS was equipped on the robot for obtaining the ground truth.

Similar to Fig. 14, Fig. 16 shows the occupancy grid map built by uspIC and the proposed algorithm. Similar to Fig. 11, Fig. 17 shows the relative localization errors among the uspIC trajectories, the proposed SLAM algorithm and the ground truth. It can also be seen clearly that the proposed algorithm outperforms the uspIC in terms of the level of map consistency with the real environment and the pose estimation accuracy.

D. Real-Time Performance

The time to obtain the dataset for both SCPBot and Ictineu AUV and the time to run the proposed algorithm RBPF-MSIS is listed in Table II. The algorithm is run on a normal laptop PC with Intel i5 CPU at 2.3 GHz and 8 GB memory. As can be seen from the table, the time to run the algorithm is much faster than the time to obtain the dataset. On the other hand, the time to run RBPF-MSIS is smaller than the time to run uspIC, which indicates that the computational complexity of RBPF-MSIS is smaller than that of uspIC.

Fig. 18 shows the time for running RBPF-MSIS algorithm after each sonar reading has been received during a full 360° of scan with SCPBot AUV. It can be seen that the execution time for the proposed algorithm for each sonar reading is much less than the time span (43 ms) for two consecutive sonar readings, and the average running time is 11 ms. Therefore, both Table II and Fig. 18 demonstrate that sonar data can be processed before a new data is received, and the proposed algorithm is suitable to be used for online applications.

V. Conclusion

This article presented a novel SLAM algorithm for AUV equipped with a MSIS sonar sensor that samples data at a slow rate. The algorithm is based on RBPF and a carefully designed sliding window-based SF module. It is able to form a complete scan each time when a single sonar reading data is obtained by incorporating the history of sonar readings. In this way, the distortion and data sparseness caused by the slow-sampling MSIS can be tackled effectively. With the generated scans, the proposed SLAM algorithm is able to produce consistent occupancy-grid maps and localize the AUV accurately. Extensive simulation and real experiment results show that the proposed algorithm outperformed other existing localization algorithm in terms of map consistency and localization accuracy.

It should be noticed that the proposed SLAM algorithm was tested off-line by using the dataset of Ictineu AUV during the real experiments. Therefore, our future work will focus on implementing the proposed algorithm on-line by using real data obtained by an AUV that operates in real world missions.

Acknowledgment

The authors would like to thank Robin Dowling and Ian Dukes for their technical support during the experiments, and to David Ribas *et al.* for making their dataset of Ictineu AUV available publicly.

References

[1] S. Venkatesan, "AUV for search and rescue at sea-an innovative approach," in *Proc. IEEE/OES Auton. Underwater Vehicles*, Nov. 2016, pp. 1–9.
[2] D. Thompson, D. Caress, H. Thomas, and D. Conlin, "MBARI mapping AUV operations in the Gulf of California 2015," in *Proc. OCEANS - MTS/IEEE Washington*, Oct. 2015, pp. 1–7.
[3] S. B. Williams *et al.*, "Monitoring of benthic reference sites: Using an autonomous underwater vehicle," *IEEE Robot. Autom. Mag.*, vol. 19, no. 1, pp. 73–84, Mar. 2012.
[4] A. Tonacci, G. Lacava, M. A. Lippa, L. Lupi, M. Cocco, and C. Domenici, "Electronic nose and AUV: A novel perspective in marine pollution monitoring," *Marine Technol. Soc. J.*, vol. 49, no. 5, pp. 18–24, 2015.
[5] A. Matos, N. Cruz, A. Martins, and F. L. Pereira, "Development and implementation of a low-cost LBL navigation system for an AUV," in *Proc. OCEANS'99 MTS/IEEE. Riding Crest into 21st Century. Conf. Exhib.*, 1999, vol. 2, pp. 774–779.
[6] K. Vickery, "Acoustic positioning systems: A practical overview of current systems," in *Proc. Workshop Auton. Underwater Vehicles*, 1998, pp. 5–17.
[7] S.-J. Li, C.-H. Tao, and G.-S. Bao, "INS/USBL underwater navigation system based on Kalman filter," *Ocean Technol.*, vol. 3, no. 14, pp. 47–50, 2008.
[8] X. Yun *et al.*, "Testing and evaluation of an integrated GPS/INS system for small AUV navigation," *IEEE J. Ocean. Eng.*, vol. 24, no. 3, pp. 396–404, Jul. 1999.
[9] A. Kim and R. M. Eustice, "Active visual SLAM for robotic area coverage: Theory and experiment," *Int. J. Robot. Res.*, vol. 34, no. 4-5, pp. 457–475, 2015.
[10] S. Hong, J. Kim, J. Pyo, and S.-C. Yu, "A robust loop-closure method for visual SLAM in unstructured seafloor environments," *Auton. Robots*, vol. 40, no. 6, pp. 1095–1109, 2016.
[11] S. Barkby, S. Williams, O. Pizarro, and M. Jakuba, "An efficient approach to bathymetric SLAM," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2009, pp. 219–224.
[12] E. Chen and J. Guo, "Real time map generation using sidescan sonar scanlines for unmanned underwater vehicles," *Ocean Eng.*, vol. 91, pp. 252–262, 2014.
[13] D. Ribas, P. Ridao, J. D. Tardós, and J. Neira, "Underwater SLAM in man-made structured environments," *J. Field Robot.*, vol. 25, no. 11-12, pp. 898–921, 2008.
[14] M. Dong, W. Chou, and B. Fang, "Underwater matching correction navigation based on geometric features using sonar point cloud data," *Scientific Program.*, vol. 2017, 2017, Art. no. 7136702.
[15] A. Burguera, Y. González, and G. Oliver, "The uspIC: Performing scan matching localization using an imaging sonar," *Sensors*, vol. 12, no. 6, pp. 7855–7885, 2012.
[16] A. Burguera, "A novel approach to register sonar data for underwater robot localization," in *Proc. Intell. Syst. Conf.*, 2017, pp. 1034–1043.
[17] A. Doucet, N. de Freitas, K. Murphy, and S. Russell, "Rao-Blackwellised particle filtering for dynamic Bayesian networks," in *Proc. Sixteenth Conf. Uncertainty Artif. Intell.*, 2000, pp. 176–183.
[18] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with Rao-Blackwellized particle filters," *IEEE Trans. Robot.*, vol. 23, no. 1, pp. 34–46, Feb. 2007.
[19] R. Smith, M. Self, and P. Cheeseman, "Estimating uncertain spatial relationships in robotics," in *Autonomous Robot Vehicles*. New York, NY, USA: Springer, 1990, pp. 167–193.
[20] M. Montemerlo, N. Roy, S. Thrun, D. Hähnel, C. Stachniss, and J. Glover, *CARMEN–The Carnegie Mellon Robot Navigation Toolkit*, (2002). [Online]. Available: http://carmen.sourceforge.net
[21] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. Cambridge, MA, USA: MIT Press, 2005, vol. 1.
[22] A. Burguera, Y. Gonzalez, and G. Oliver, "Probabilistic sonar scan matching for robust localization," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2007, pp. 3154–3160.
[23] *3D simulator-Gazebo*, http://www.ros.org/wiki/gazebo, [Online; accessed 28-November-2018].
[24] "Robot Operating System," Accessed: Nov. 28, 2018. [Online]. Available: http://www.ros.org/wiki/
[25] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2012, pp. 573–580.

Ling Chen received both the B.Eng. and M.Eng. degrees in control science and engineering from Central South University, Changsha, China, in 2008 and 2010, respectively, and the Ph.D. degree in computing and electronics engineering from the University of Essex, Colchester, U.K., in 2014.

He is currently a Lecturer with the School of Engineering and Design, Hunan Normal University, Changsha, China. His research interests include underwater simultaneous localization and mapping (SLAM), visual SLAM and robot localization in general.

Aolei Yang received the B.Sc. degree from the Hubei University of Technology, Wuhan, China, in 2004, the M.Sc. degree from Shanghai University, Shanghai, China, in 2009, and Ph.D. degree from Queens University Belfast, Belfast, U.K., in 2012.

He is currently an Associate Professor with the School of Mechatronic Engineering and Automation, Shanghai University, Shanghai, China. He is the author of over 40 peer-reviewed journal and conference papers. His current research interests include robotics, image processing and vision learning, and cooperative control of multiagents.

Dr. Yang is the Secretary General of the Embedded Instrument and System Technology Branch of China Instrument and Control Society(CIS), and the Deputy Secretary General of Shanghai Instrument Society.

Huosheng Hu (M'94–SM'01) received the M.Sc. degree in industrial automation from Central South University, Changsha, China, in 1982, and the Ph.D. degree in robotics from the University of Oxford, Oxford, U.K., in 1993.

He is currently a Professor with the School of Computer Science and Electronic Engineering, University of Essex, Colchester, U.K., leading the robotics research. He has authored over 500 papers in the areas of robotics, humanCrobot interaction, embedded systems, mechatronics, and pervasive computing.

Prof. Hu is a Founding Member of the IEEE Robotics and Automation Society Technical Committee on Networked Robots, a Fellow of IET and InstMC. He has been the Program Chair and a member of the Advisory or Organizing Committee for many international conferences, such as the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE International Conference on Robotics and Automation (ICRA), IEEE International Conference on Mechatronics and Automation (ICMA), IEEE International Conference on Robotics and Biomimetics (ROBIO) and IEEE International Conference on Information and Automation (ICIA). He currently serves as an Editor-in-Chief of the *International Journal of Automation and Computing*, Editor-in-Chief of *MDPI Robotics journals*, and an Executive Editor of the *International Journal of Mechatronics and Automation*.

Wasif Naeem received the B.Eng. degree from NED University, Karachi, Pakistan, the M.Sc. degrees from King Fahd University, Dhahran, Saudi Arabia, in 1998 and 2001, respectively, both in electrical engineering, and the Ph.D. degree in mechanical and marine engineering from the University of Plymouth, Plymouth, U.K., in 2004.

He is currently a Senior Lecturer with the School of Electronics, Electrical Engineering, and Computer Science, Queens University Belfast, Belfast, U.K. He has authored over 80 peer-reviewed journal and conference papers and three IET book chapters. His current research interests include optimal and robust control, system identification, multivehicle formation control, and systems engineering.

Dr. Naeem was a recipient of the Michael Richey Medal and the Denny Medal in 2008 and 2010 by the Royal Institute of Navigation and the Institution of Marine Engineering, Science and Technology, respectively. He is a member of the International Federation of Automatic Control (IFAC) technical committee on marine systems and has served on the committee of a number of international conferences.