IET Communications

Research Article

IET Journals
The Institution of Engineering and Technology

Target tracking based on improved square root cubature particle filter via underwater wireless sensor networks

ISSN 1751-8628
Received on 13th March 2018
Revised 10th October 2018
Accepted on 30th January 2019
E-First on 19th March 2019
doi: 10.1049/iet-com.2018.5097
www.ietdl.org

Hailin Feng[1], *Zhiwei Cai*[1] ✉

[1]*School of Mathematics and Statistics, Xidian University, Xi'an Shaanxi, People's Republic of China*

✉ *E-mail: ZhiweiCai_XD@126.com*

Abstract: Target tracking in underwater wireless sensor networks (UWSNs) has two fundamental issues that tracking accuracy and energy consumption. Although the application of particle filter in target tracking is considered in recent years, degeneracy phenomenon and particle impoverishment always restrict its capacity and application. This paper proposes an improved square root cubature particle filter (ISRCPF) to improve tracking accuracy. The authors employ self-adaptive artificial fish swarm algorithm (AFSA) to optimise the particles, which makes the particles move towards to high likelihood region and maintains the diversity of the particles. Moreover, a sensor selection scheme is provided, which reduces energy consumption of the network by exactly waking up four sensor nodes at each time, while preserving tracking accuracy. Additionally, the authors propose a novel fusion method called similarity fusion method (SFM) to fuse local estimates together and then obtain a better result for distributed fusion architecture (DFA). The simulation results demonstrate that the proposed methods have superior performance.

1 Introduction

Nowadays, underwater target tracking using underwater wireless sensor networks (UWSNs) is becoming an essential part in both military and non-military applications [1–3]. On the one hand, UWSNs interact in the marine environment and have broader applications in data acquisition, monitoring, navigation, and undersea exploration [4, 5]. On the other, using sonar arrays is difficult and even impractical for completing such missions because they should be installed or carried by a ship or submarine [6]. However, due to using sound waves for communication and existence of uncertainty in the system dynamics and measurements, it becomes difficult to obtain the exact path of the target [7].

There are plenty of studies devoted to tracking methods for the sake of tracking the target. In [8], Isbitiren and Akan proposed the three-dimensional underwater target tracking (3DUT) algorithm. Trilateration is used in this algorithm to continuously calculate the location of the target. However, this method cannot provide high accuracy due to not having prediction steps. An adaptive Kalman filter is proposed by Li *et al.* [9] based on the online approximation of the process noise variance. Yet, this plan also has limitations that it only can be used in linear system and performs tracking in 2D space. In [7], Extended Kalman filter (EKF) and Unscented Kalman filter (EKF) are applied to estimate target path in 3D space compared with geometric approach. In this research, EKF and UKF perform better than geometric approach, however, the accuracy will decrease when the state dimensionality is relatively high [10]. In [11], time delay neural network-based prediction algorithm is proposed for target tracking in WSNs, which outperforms Kalman filter and interacting multiple model filter. However, this plan only performs 2D tracking and does not consider the non-linear systems. Chen *et al.* proposed radial basis function neural network-based prediction method for target tracking in chain-type WSNs [12]. Yet, this method only uses one tracking target node, which cannot give full play to the advantages of WSNs. Particle filter (PF) is applicable for target tracking by using a set of particles to effectively represent the prior and posterior likelihood of states [13, 14]. In [15], two algorithms are proposed to tracking a moving target with UWSNs by using PF. Simulation results show that the first algorithm is more accurate than EKF, while the second decreases the cost of communications but has lower accuracy in tracking. However, PF has common problems that are the phenomenon of degeneracy and particle impoverishment [16]. In [17], auxiliary particle filter (APF) is used to track a highly manoeuvring target. When the process noise is large, the use of APF will degrade the performance. In [18], the EKF is used to generate the importance distribution for target tracking based on PF. However, approximation of linearisation limits the performance improvement. In [19], an improved unscented particle filter is proposed by using UKF to obtain the importance distribution. Yet, the weights of UKF are negative when the dimension of the system is more than three, which will cause the divergence of the filter.

Another important issue in UWSNs is minimising energy consumption. Since the energy of sensor nodes are supplied by batteries, and when they are exhausted, the substitution of energy supply is impossible [20]. Centralised fusion architecture (CFA) reduces the reliability and fault-tolerance of systems and consume more energy in fusion centre. Neighbouring nodes of fusion centre should route data towards this centre and this causes unbalanced consumption of energy in networks. Therefore, the energy of fusion centre and its neighbour nodes consume rapidly and cause the life time of network to be decreased. Fortunately, distributed fusion architecture (DFA) can overcome this limitation [21]. In [22], a local node selection scheme is proposed which increases energy efficiency by waking up only a small part of nodes at each time. However, the tacking accuracy cannot be guaranteed when less than four nodes are waked up, while energy consumption will be high when more than four nodes are waked up. Hare *et al.* proposed a decentralised method for sensor scheduling. [23] It allows dynamic time and space clustering for target tracking with effective energy in a sensor network. However, it is impractical to utilise this method in 3D space. In [20], an energy-efficient filter is proposed. Under the distributed fusion framework, local sensors should not send their weak information to the fusion centre if their measurement residuals are smaller than the pre-given threshold.

This paper proposes an improved square root cubature particle filter (ISRCPF) algorithm for underwater target tracking. It uses square root cubature Kalman filter to generate a better proposal distribution within PF framework and employs self-adaptive artificial fish swarm algorithm (AFSA) to optimise the particles. Moreover, the sensor selection scheme is utilised to reduce energy consumption of the network, while preserving tracking accuracy. In addition, since multi-sensors observe a same target and track it in DFA of the network, the estimates should be fused to get a better

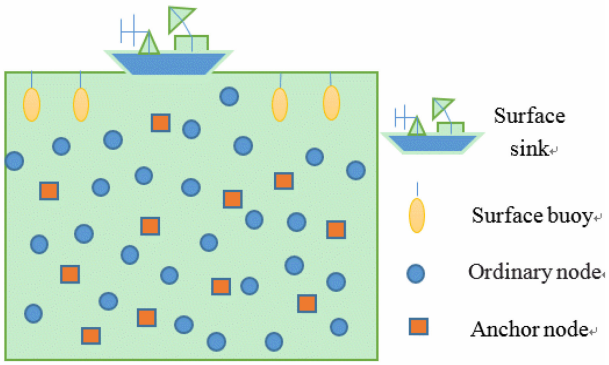

Fig. 1 *Underwater network architecture*

result. Therefore, we propose a novel estimate fusion method based on the similarity between estimates to improve the stability and accuracy of estimation.

The remaining of this paper proceeds as follows. Section 2 describes the problem by introducing network model as well as system model. In Section 3, a target tracking algorithm is proposed by embedding self-adaptive AFSA into square root cubature particle filter. Section 4 presents the sensor selection scheme and a novel estimate fusion method in DFA. In Section 5, simulation results are presented and the analysis of the comparison are accomplished. Finally, Section 6 draws conclusions of this study.

2 Problem formulation

2.1 Network model

Fig. 1 shows a typical architecture of UWSNs used for 3DUT tracking. It consists of three types of nodes: surface buoys, anchor nodes and ordinary nodes. Surface buoys drifting on the water surface are usually equipped with global position system to get their absolute locations. Anchor nodes are those who can get their absolute positions by contacting the surface buoys directly. Ordinary nodes cannot directly contact the surface buoys because of the cost or other constraints, but they can communicate with the anchor nodes to estimate their positions. Anchor nodes and ordinary nodes are randomly deployed over the determinate area in the 3D environment. A lot of studies already exist for the location of UWSNs [24, 25]. Here, locating the nodes will not be considered, instead, we mainly address the problem of underwater target tracking with UWSNs.

2.2 System model

Assume that there is only one target moving within the known region. At time k, the state vector of the target in 3D Cartesian coordinate is defined as

$$\boldsymbol{X}_k = [x_k \quad \dot{x}_k \quad y_k \quad \dot{y}_k \quad z_k \quad \dot{z}_k], \tag{1}$$

whose components are x position, x velocity, y position, y velocity, z position, z velocity. The state dynamic equation is given by

$$\boldsymbol{X}_k = \boldsymbol{F}_k \boldsymbol{X}_{k-1} + \boldsymbol{w}_k, \tag{2}$$

where $\boldsymbol{F}_k$ is the state transition matrix, and $\boldsymbol{w}_k$ is the independent process noise with zero-mean, white, Gaussian probability distribution $N(0, \boldsymbol{Q}_k)$.

According to [22], range-only measurement model is more practical than linear measurement model which assumes that one sensor node can locate the target. Therefore, the range-only measurement model is utilised here. Assume that there are M wireless acoustic sensors in UWSNs. The position of sensor i is denoted by $\boldsymbol{S}_i = (x_i^s, y_i^s, z_i^s), i = 1, 2, \ldots, M$. The distance between sensor i and the target is measured by pluses and calculating the time-of-arrival (TOA) of pluses and echoes. Therefore, the measurement model for sensor i at time k is given by

$$Z_{i,k}(k) = \sqrt{(x_k - x_i^s)^2 + (y_k - y_i^s)^2 + (z_k - z_i^s)^2} + v_{i,k} \tag{3}$$

where (x_k, y_k, z_k) is the location of the target at time k, and $v_{i,k}$ is the measurement noise with zero-mean, white, Gaussian probability distribution $N(0, R_{i,k})$.

3 Target tracking algorithms

In this section, we start with a review of square root cubature particle filter algorithm. Then, the AFSA is introduced, which will be utilised to optimise square root cubature particle filter algorithm. Finally, we present the ISRCPF algorithm by using the self-adaptive AFSA.

3.1 Square root cubature particle filter

Simplify the state dynamic model and the measurement model in Section 2.2 as follows.

$$\begin{aligned} \boldsymbol{x}_k &= f(\boldsymbol{x}_{k-1}) + \boldsymbol{w}_{k-1} \\ z_k &= h(x_k) + v_k \end{aligned} \tag{4}$$

$\{\boldsymbol{x}_k^{(i)}, i = 1, 2, \ldots, N\}$ with associated weights $\{w_k^{(i)}, i = 1, 2, \ldots, N\}$ is a set of particles randomly sampled from posterior probability density function $p(\boldsymbol{x}_k | \boldsymbol{z}_{1:k})$, N is the number of particles and $\sum_{i=1}^{N} w_k^{(i)} = 1$. Therefore, the posterior probability density at time k can be weighted discretely as follows:

$$p(\boldsymbol{x}_k | \boldsymbol{z}_{1:k}) = \sum_{k=1}^{N} w_t^{(i)} \delta\left(\boldsymbol{x}_k - \boldsymbol{x}_k^{(i)}\right) \tag{5}$$

where $w_t^{(i)}$ needs to be determined from important sampling methods and $\delta(\cdot)$ is a Dirac's function. It is usually very difficult to sample directly from the posterior probability distribution. The conventional method is to indirectly sample from a distribution $q\left(\boldsymbol{x}_k^{(i)} \middle| \boldsymbol{x}_{0:k-1}^{(i)}, \boldsymbol{z}_{1:k}\right)$ with a known probability density distribution function that is easy to sample. Thus the weight can be represented as

$$w_k^{(i)} \propto \frac{p\left(z_k \middle| \boldsymbol{x}_k^{(i)}\right) p\left(\boldsymbol{x}_k^{(i)} \middle| \boldsymbol{x}_{k-1}^{(i)}\right)}{q\left(\boldsymbol{x}_k^{(i)} \middle| \boldsymbol{x}_{0:k-1}^{(i)}, \boldsymbol{z}_{1:k}\right)} \tag{6}$$

For the SRCPF algorithm, in the sampling phase, the SRCKF is used to calculate the mean and the covariance of each particle, which approximates the posterior density at each moment according to the following formula:

$$p(\boldsymbol{x}_k | \boldsymbol{z}_{1:k}) \simeq N\left(\hat{\boldsymbol{x}}_k, \hat{\boldsymbol{P}}_k\right) \tag{7}$$

where $\hat{\boldsymbol{x}}_k$ and $\hat{\boldsymbol{P}}_k$ are the state estimate and the covariance estimate at time k, respectively. Under the framework of PF, the particles are updated by the SRCKF, and the approximate posterior density obtained is taken as the importance density function

$$q\left(\boldsymbol{x}_k^{(i)} \middle| \boldsymbol{x}_{0:k-1}^{(i)}, \boldsymbol{z}_{1:k}\right) = N\left(\hat{\boldsymbol{x}}_k^{(i)}, \boldsymbol{P}_k^{(i)}\right) \tag{8}$$

Then, the new particles are generated from the importance distribution and the weights are updated, and finally the particle set is resampled. The SRCPF algorithm is described as follows:

(1) Initialisation. $k = 0$ is set, sample N particles and represent them as $\{x_0^{(i)}, i = 1, 2, \ldots, N\}$ from the prior distribution $p(x_0)$, and obtain a square root $S_0^{(i)}$ of the covariance $P_0^{(i)}$.

(2) Assume the particles and the square root of the covariance at time $k-1 (k = 1, 2, \ldots)$ are $\left(\boldsymbol{x}_{k-1}^{(i)}, \boldsymbol{S}_{k-1}^{(i)}\right), i = 1, 2, \ldots, N$. Carry out the following steps.

(a) Importance sampling step

(i) Update the particles using SRCKF

Evaluate cubature points and propagate the state equation

$$\boldsymbol{X}_{k-1,j}^{(i)} = \boldsymbol{S}_{k-1}^{(i)}\xi_j + \boldsymbol{x}_{k-1}^{(i)}, \quad \boldsymbol{X}_{k,j}^{(i)*} = f\left(\boldsymbol{X}_{k-1,j}^{(i)}\right) \tag{9}$$

where the cubature points are denoted by $\boldsymbol{\xi}_j = \sqrt{m/2}[1]_j$, and weights $\omega_j = 1/m, j = 1, \ldots, m = 2n_x$.

Estimate the predicted particles and the square root of the predicted covariance

$$\bar{\boldsymbol{x}}_k^{(i)} = \sum_{j=1}^{m} \omega_j \boldsymbol{X}_{k,j}^{(i)*}, \quad \bar{\boldsymbol{S}}_k^{(i)} = Tria\left(\left[\boldsymbol{\chi}_{k,j}^{(i)*} \quad \boldsymbol{S}_{Q,k-1}\right]\right) \tag{10}$$

where $\boldsymbol{Q}_k = \boldsymbol{S}_{Q,k-1}\boldsymbol{S}_{Q,k-1}^{\mathrm{T}}$ and $Tria()$ is denoted as a general triangularisation algorithm. The weighted, centred matrix is defined as

$$\boldsymbol{\chi}_{k,j}^{(i)*} = \frac{1}{\sqrt{m}}\left[\boldsymbol{X}_{k,1}^{(i)*} - \bar{\boldsymbol{x}}_k^{(i)} \quad \boldsymbol{X}_{k,2}^{(i)*} - \bar{\boldsymbol{x}}_k^{(i)}, \ldots, \boldsymbol{X}_{k,m}^{(i)*} - \bar{\boldsymbol{x}}_k^{(i)}\right] \tag{11}$$

Evaluate the predicted cubature points and propagate the measurement equation

$$\boldsymbol{X}_{k,j}^{(i)} = \bar{\boldsymbol{S}}_k^{(i)}\xi_j + \bar{\boldsymbol{x}}_k^{(i)}, \quad Z_{k,j}^{(i)} = h\left(\boldsymbol{X}_{k,j}^{(i)}\right). \tag{12}$$

Estimate the predicted measurement, square root of the innovation covariance and cross-covariance

$$\begin{aligned} \bar{z}_k^{(i)} &= \sum_{j=1}^{m} \omega_j Z_{k,j}^{(i)}, \quad \boldsymbol{S}_{zz,k}^{(i)} = Tria\left(\left[\boldsymbol{\gamma}_k^{(i)} \quad \boldsymbol{S}_{R,k}\right]\right), \\ \boldsymbol{P}_{xz,k}^{(i)} &= \boldsymbol{\chi}_k^{(i)}\boldsymbol{\gamma}_k^{(i)\mathrm{T}}. \end{aligned} \tag{13}$$

where $\boldsymbol{R}_k = \boldsymbol{S}_{R,k}\boldsymbol{S}_{R,k}^{\mathrm{T}}$ and the weighted, centred matrix $\boldsymbol{\gamma}_k^{(i)}$ and $\boldsymbol{\chi}_k^{(i)}$ are defined as

$$\begin{aligned} \boldsymbol{\gamma}_k^{(i)} &= \frac{1}{\sqrt{m}}\left[Z_{k,1}^{(i)} - \bar{z}_k^{(i)} \quad Z_{k,2}^{(i)} - \bar{z}_k^{(i)}, \ldots, Z_{k,m}^{(i)} - \bar{z}_k^{(i)}\right] \\ \boldsymbol{\chi}_k^{(i)} &= \frac{1}{\sqrt{m}}\left[\boldsymbol{X}_{k,1}^{(i)} - \bar{\boldsymbol{x}}_k^{(i)} \quad \boldsymbol{X}_{k,2}^{(i)} - \bar{\boldsymbol{x}}_k^{(i)}, \ldots, \boldsymbol{X}_{k,m}^{(i)} - \bar{\boldsymbol{x}}_k^{(i)}\right] \end{aligned} \tag{14}$$

Evaluate the gain, estimated state and square root of estimated covariance

$$\begin{aligned} W_k^{(i)} &= \boldsymbol{P}_{xz,k}^{(i)}\left(\boldsymbol{S}_{zz,k}^{(i)}\boldsymbol{S}_{zz,k}^{(i)\mathrm{T}}\right)^{-1} \\ \hat{\boldsymbol{x}}_k^{(i)} &= \bar{\boldsymbol{x}}_k^{(i)} + W_k^{(i)}\left(z_k - \bar{z}_k^{(i)}\right) \\ \boldsymbol{S}_k^{(i)} &= Tria\left(\left[\boldsymbol{\chi}_k^{(i)} - W_k^{(i)}\boldsymbol{\gamma}_k^{(i)} \quad W_k^{(i)}\boldsymbol{S}_{R,k}\right]\right). \end{aligned} \tag{15}$$

Sample $\boldsymbol{x}_k^{(i)} \sim q\left(\boldsymbol{x}_k^{(i)} \middle| \boldsymbol{x}_{0:k-1}^{(i)}, z_{1:k}\right) = N\left(\hat{\boldsymbol{x}}_k^{(i)}, \boldsymbol{S}_k^{(i)}\boldsymbol{S}_k^{(i)\mathrm{T}}\right)$ and obtain particles $\left(\boldsymbol{x}_k^{(i)}, \boldsymbol{S}_k^{(i)}\right)$.

(ii) Evaluate the importance weights of particles and normalise them

$$\begin{aligned} w_k^{(i)} &= w_{k-1}^{(i)} \frac{p\left(z_k \middle| \hat{\boldsymbol{x}}_k^{(i)}\right) p\left(\hat{\boldsymbol{x}}_k^{(i)} \middle| \hat{\boldsymbol{x}}_{k-1}^{(i)}\right)}{q\left(\hat{\boldsymbol{x}}_k^{(i)} \middle| \hat{\boldsymbol{x}}_{0:k-1}^{(i)}, z_{1:k}\right)} \\ \bar{w}_k^{(i)} &= w_k^{(i)} / \sum_{j=1}^{N} w_k^{(j)}. \end{aligned} \tag{16}$$

where $p\left(z_k \middle| \hat{\boldsymbol{x}}_k^{(i)}\right)$ is likelihood function and $p\left(\hat{\boldsymbol{x}}_k^{(i)} \middle| \hat{\boldsymbol{x}}_{k-1}^{(i)}\right)$ is state transition function.

(b) *Resampling step:* Multiply (Suppress) particles with high (low) weights to obtain N random particles $x_k^{(i)}$ with weights N^{-1}.

(3) Each particle and the square root of covariance are $\left(\boldsymbol{x}_k^{(i)}, \boldsymbol{S}_k^{(i)}\right)$. Estimate the mean state and covariance at time k.

$$\begin{aligned} \hat{\boldsymbol{x}}_k &= \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{x}_k^{(i)}, \\ \boldsymbol{P}_k &= \frac{1}{N}\sum_{i=1}^{N} (\boldsymbol{x}_k^{(i)} - \hat{\boldsymbol{x}}_k)(\boldsymbol{x}_k^{(i)} - \hat{\boldsymbol{x}}_k)^{\mathrm{T}}. \end{aligned} \tag{17}$$

The SRCPF algorithm applying SRCKF gets a better importance distribution. However, the phenomenon of degeneracy still exists, which means that the variance of weight of particles increases with time. In other words, the information always focuses on the minority of particles with higher weights. Moreover, the number cubature points of SCKF is small, which may not enough to represent complex distributions. Besides, the SRCPF still uses the traditional resampling method, which will lead to the problem of particle impoverishment. As it duplicates a few particles with large weights and discards majority of particles with low weights, which is not conducive to the diversity of particles. Therefore, it is necessary to improve the performance of the SRCPF algorithm. We propose an ISRCPF algorithm based on the self-adaptive AFSA.

3.2 Artificial fish swarm algorithm

AFSA is a stochastic searching optimisation algorithm based on simulating fish swarm's ecological behaviour in the natural environment [26]. It has been widely applied in many fields like pattern recognition, parameter estimation, and neural networks because of the merits of rapid and global optimisation, insensitivity to initial values and selections of parameters, robustness, easy operation, and so on [27]. Fish swarm generally has four kinds of typical behaviour including prey behaviour, swarm behaviour, follow behaviour, and random behaviour. We mainly introduce the first two behaviours that will be put into use for optimising the SRCPF algorithm.

Suppose that the individual state of artificial fish (AF) is denoted by a vector $\boldsymbol{X} = (x_1, x_2, \ldots, x_n)$, where $x_i (i = 1, 2, \ldots, n)$ is the variable to be searched for optimal. The concentration of food in the current position of AF is denoted by $Y = f(\boldsymbol{X})$, where Y is the value of objection function. The distance between two AF individuals can be expressed as $d_{ij} = \| \boldsymbol{X}_i - \boldsymbol{X}_j \|$. A positive constant v represents visual distance of AF. The description of behaviours of AF is described as follows.

Prey behaviour: Let $\boldsymbol{X}_i$ denote the current state of the AF and select a state $\boldsymbol{X}_j$ randomly within the scope of perception ($d_{ij} < v$). We discuss the maximum problem here. If $Y_i < Y_j$, $\boldsymbol{X}_i$ moves a step towards $\boldsymbol{X}_j$, otherwise, we select randomly a state $\boldsymbol{X}_j$ again and justify whether it satisfies the above requirement. If the requirement cannot be satisfied after trying several times, $\boldsymbol{X}_i$ moves a step randomly. Showing briefly the process of the prey behaviour is following.

$$\begin{aligned} &\text{if } Y_i < Y_j, \quad \boldsymbol{X}_{i,\text{next}} = \boldsymbol{X}_i + r*s*\frac{\boldsymbol{X}_j - \boldsymbol{X}_i}{\| \boldsymbol{X}_j - \boldsymbol{X}_i \|} \\ &\text{else}, \quad \boldsymbol{X}_{i,\text{next}} = \boldsymbol{X}_i + \boldsymbol{r}*s \end{aligned} \tag{18}$$

where s is the largest step length of the AF, r is a random number ranging from zero to one, and $\boldsymbol{r} \in U(0, 1)$ is a random vector with the same dimension of $\boldsymbol{X}_i$.

Swarm behaviour: Let $\boldsymbol{X}_i$ denote the current state of the AF and n_f denote the number of swarm-fellows within the scope of perception ($d_{ij} < v$), $\boldsymbol{X}_c = (1/n_f)\sum_{j=1}^{n_f} \boldsymbol{X}_j$ denotes the centre of the fellows, Y_c denotes the food concentration of $\boldsymbol{X}_c$, and $\delta(0 < \delta < 1)$

1. Initialization $k = 0$

For $i = 1,2,\ldots,N$ **do**

Draw $\boldsymbol{x}_0^{(i)} \backsim p(\boldsymbol{x}_0)$; Set $w_0^{(i)} = 1/N, \hat{\boldsymbol{x}}_0^{(i)} = E(\boldsymbol{x}_0^{(i)})$ and $\boldsymbol{P}_0^{(i)} = E\left[\left(\boldsymbol{x}_0^{(i)} - \hat{\boldsymbol{x}}_0^{(i)}\right)\left(\boldsymbol{x}_0^{(i)} - \hat{\boldsymbol{x}}_0^{(i)}\right)^T\right]$;

Let $S_0^{(i)}$ be the square root of $P_0^{(i)}$;

2. For $k = 1,2,\ldots$ **do**

2.1 Importance Sampling:

For $i = 1,2,\ldots,N$ **do**

Update particles with the SRCKF using Eq.(9)-Eq.(15) and draw $\boldsymbol{x}_k^{(i)} \sim q\left(\boldsymbol{x}_k^{(i)} \middle| \boldsymbol{x}_{0:k-1}^{(i)}, \boldsymbol{z}_{1:k}\right) = N(\hat{\boldsymbol{x}}_k^{(i)}, S_k^{(i)} S_k^{(i)^T})$;

2.2 Optimizing particles by SAFSA:

For $i = 1,2,\ldots,N$ **do**

Set $\boldsymbol{x}_k^{(i_0)} = \boldsymbol{x}_k^{(i)}, y_{i_0} = y_i$;

For $m = 1,2,\ldots,D$ **do**

2.2.1 Prey behaviour:

if $y_{i_{m-1}} < y_j$

$$\boldsymbol{x}_k^{(i_m)} = \boldsymbol{x}_k^{(i_{m-1})} + r * s * \frac{\boldsymbol{x}_k^{(j)} - \boldsymbol{x}_k^{(i_{m-1})}}{\left\| \boldsymbol{x}_k^{(j)} - \boldsymbol{x}_k^{(i_{m-1})} \right\|};$$

else $\boldsymbol{x}_k^{(i_m)} = \boldsymbol{x}_k^{(i_{m-1})} + r * s$;

$s = \alpha \cdot s$;

2.2.2 Swarm behaviour:

if $y_{ic}/|A_v| > \delta \cdot y_{i_{m-1}}$ ***and*** $y_{i_{m-1}} < y_{ic}$

$$\boldsymbol{x}_k^{(i_m)} = \boldsymbol{x}_k^{(i_{m-1})} + r * s * \frac{\boldsymbol{x}_k^{(ic)} - \boldsymbol{x}_k^{(i_{m-1})}}{\left\| \boldsymbol{x}_k^{(ic)} - \boldsymbol{x}_k^{(i_{m-1})} \right\|};$$

else *turn to pery behaviour*;

$v = v_0 - v_0 \cdot m/D$;

2.3 Importance weights updating:

Calculate $w_k^{(i)} = w_{k-1}^{(i)} \dfrac{p\left(z_k \middle| \boldsymbol{x}_k^{(i)}\right) p\left(\boldsymbol{x}_k^{(i)} \middle| \boldsymbol{x}_{k-1}^{(i-1)}\right)}{q\left(\boldsymbol{x}_k^{(i)} \middle| \boldsymbol{x}_{0:k-1}^{(i)}, \boldsymbol{z}_{1:k}\right)}$;

Normalize $\overline{w}_k^{(i)} = w_k^{(i)} / \sum_{j=1}^{N} w_k^{(j)}$;

2.4 Estimate the mean state and covariance:

$\hat{\boldsymbol{x}}_k = \sum_{i=1}^{N} w_k^{(i)} \boldsymbol{x}_k^{(i)}$;

$\hat{\boldsymbol{P}}_k = \sum_{i=1}^{N} w_k^{(i)} (\boldsymbol{x}_k^{(i)} - \hat{\boldsymbol{x}}_k)(\boldsymbol{x}_k^{(i)} - \hat{\boldsymbol{x}}_k)^T$;

Fig. 2 *ISRCPF algorithm*

is the crowding factor. If $Y_c/n_f > \delta Y_i$ and $Y_i < Y_c$, which indicates that the centre of the fellows exists abundant food and is not quite crowded. Thus, $\boldsymbol{X}_i$ moves a step towards the centre $\boldsymbol{X}_c$. Otherwise, it executes the prey behaviour. Showing briefly the process of the swarm behaviour is following.

$$\begin{aligned} &\text{if } Y_c/n_f > \delta \cdot Y_i \text{ and } Y_i < Y_c, \\ &\boldsymbol{X}_{i,\text{next}} = \boldsymbol{X}_i + r^* s^* \frac{\boldsymbol{X}_c - \boldsymbol{X}_i}{\| \boldsymbol{X}_c - \boldsymbol{X}_i \|} \\ &\text{else,} \quad \text{turn to pery behaviour} \end{aligned} \tag{19}$$

3.3 Improved square root cubature particle filter

Here, the SRCPF algorithm is combined with AFSA. According to Section 3.2, the AF will always randomly search the area with high food concentration by constantly updating their own positions. So we consider the particles in SRCPF as the AF in AFSA, objective function as posterior probability density function and the likelihood degree as the food concentration. Utilising prey behaviour and swarm behaviour of AFSF to optimise the particles. With this combination, moving the AF to be close to the optimal AF brings about driving the particles move towards the high likelihood area. Thus, the problem of particles impoverishment will be solved on a large scale.

In the SRCPF algorithm optimised by AFSA, if the fixed step length and the fixed visual distance are adopted, the estimation performance may not be perfect. This is because in the case of fixed step length, the larger step length is beneficial to convergence as soon as possible, but it is easy to miss the global optimal solution; Small step length is in favour of local searching, while it is slow to find the global optimal solution and easy to fall into local extremum. The same thing will happen with the fixed visual distance. Therefore, here the self-adaptive AFSA (SAFSA) with dynamic strategy is employed.

For the step length and visual distance, we let $s = \alpha \cdot s, v = v_0 - v_0 \cdot \text{iter}/D$, respectively, where $\alpha (0 < \alpha < 1)$ is attenuation factor, v_0 is the initial visual distance, *iter* is the number of iteration, and D is the total number of iteration. Thus, in the early stage of the iteration, the algorithm with large step length and large visual distance has a strong ability of global search. In the late stage, the algorithm self-adaptively reduce the step length and visual distance, so as to enhance the capability of local search. As a result, the performance of the SRCPF algorithm will be improved and we refer to the new algorithm as ISRCPF.

Define the likelihood function as the objective function:

$$y = \frac{1}{\sqrt{2\pi \det(R_k)}} \exp\left[-\frac{1}{2}\left(z_k - \hat{z}_k^{(i)}\right)^{\mathrm{T}} R_k^{-1}\left(z_k - \hat{z}_k^{(i)}\right)\right] \tag{20}$$

where $\hat{z}_k^{(i)} = h\left(\boldsymbol{x}_k^{(i)}\right)$ is the prediction measurement of particle $\boldsymbol{x}_k^{(i)}$. For prey behaviour, let $\boldsymbol{x}_k^{(i_m)}$ be the result of iterating $m (m = 1, 2, \ldots, D)$ times for $\boldsymbol{x}_k^{(i)}$, y_i, y_j be values of objection function obtained from (20). For swarm behaviour, let A_v be the set of predictions of measurements which is within v for $\hat{z}_k^{(i)}$, $|A_v|$ be the cardinality of A_v, so the centre of A_v is $\hat{z}_k^{(ic)} = (1/|A_v|) \sum_{j \neq i}^{|A_v|} \hat{z}_k^{(j)}$, $\boldsymbol{x}_k^{(ic)} = (1/|A_v|) \sum_{j=1}^{|A_v|} \boldsymbol{x}_k^{(i)}$. Detailed pseudo-code of ISRCPF algorithm is given below (Fig. 2):

4 Sensor selection scheme and distributed estimation fusion

4.1 Sensor selection scheme

According to [8], four sensor nodes are employed simultaneously to locate the target in 3D networks. Intuitively, the accuracy of tracking improves as the number of sensor nodes increases. However, the use of more than four nodes may marginally improve the performance of tracking at the expense of enhancing energy consumption and an increased computational cost. Therefore, here, we dynamically select four sensors to complete the target tracking at every tracking moment from candidate sensors which are supposed to be in a specified radius around the target.

Fig. 3 displays the basic idea of the sensor selection scheme in UWSNs. As we can see, the fusion node at time k predicts the position of the target for time $k + 1$, and the great solid line circle with the centre of the predicted position and the specified radius set before tracking contains five candidate sensor nodes for the tracking mission. We should select four nodes from the five, and there will be ten options because of the combination in mathematics. Since the longer the acoustic wave propagation distance, the more serious the signal attenuation and the greater the time delay. It will lead to that the information is too weak and measurements become inaccurate. Thus, tracking accuracy will be seriously affected. Therefore, in our scheme, four nodes closest to the predicted position of target will wake up and the rest will stay sleeping. In addition, the closest one of the four will become the new fusion centre node at time $k + 1$.

4.2 Distributed estimate fusion

As introduced in the previous part, there are some different local sensor nodes tracking the same target at the same time and thus different measurements are acquired. To obtain more accurate

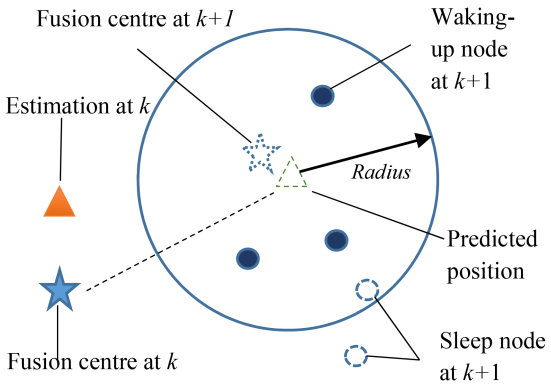

Fig. 3 *Sensor selection scheme*

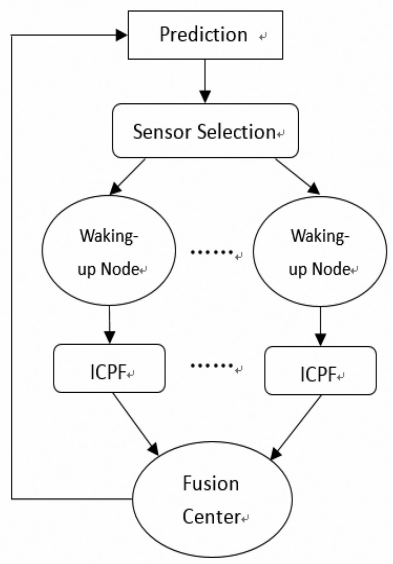

Fig. 4 *Flow chart of tracking underwater target*

estimates of target states, it is extremely necessary to fuse together the information coming from different sensor nodes. DFA and CFA are the two common fusion architectures. Considering sensor nodes' limited computing ability and resource in UWSNs, DFA is utilised preferentially instead of CFA. In DFA, each sensor has its own processing unit to run the filter (ISRCPF, here) based on the current measurements. Then, the fusion centre fuses local estimates from each local sensor together by using distributed estimate fusion method. Therefore, we propose a novel-distributed estimate fusion method based on the similarity between the local estimates to get a better result. We name this method as similarity fusion method (SFM).

Since in reality we never know the true state of the target. We can only get estimates of its state, so we start with these estimates to make them together. Although all sensors are identical, their estimates are different due to some random factors. There are bound to be some conflicts between these estimates and they cannot be treated equally. As if there is one estimate that is quite different from the others, but we treat them equally, this will lead to very inaccurate fusion results. Therefore, in our method, we treat each estimate as an evidence, if some evidence is supported by the others, it means that it is more credible, and its weight is also larger, thus the impact on the fusion result is also greater, On the contrary, if there are big conflicts between some evidence and the others, this evidence has lower credibility, and its weight is lower, thus the impact on the fusion result is smaller.

Since each estimate is essentially a vector, the similarity between vectors can be measured by the cosine of angle between them. While the cosine only measures the angle between vectors, which is not enough. So our method considers both the cosine of angle and the distance between them. Assume $\boldsymbol{X}_k^i, \boldsymbol{P}_k^i, i = 1, 2, \ldots, H$ are the state estimate and covariance estimate of local sensor i at time k, where H is the number of local sensors in DFA. We define similarity between two estimates.

Definition: Let $\boldsymbol{X}_k^i, \boldsymbol{X}_k^j$ be two estimates, then the similarity of them is

$$\begin{aligned} \mathrm{SIM}\left(\boldsymbol{X}_k^i, \boldsymbol{X}_k^j\right) &= \cos(\varphi) \cdot \exp\left[-\frac{\left\| \boldsymbol{X}_k^i - \boldsymbol{X}_k^j \right\|^2}{2}\right] \\ &= \frac{\boldsymbol{X}_k^i \cdot \boldsymbol{X}_k^j}{\left\| \boldsymbol{X}_k^i \right\| \cdot \left\| \boldsymbol{X}_k^j \right\|} \cdot \exp\left[-\frac{\left\| \boldsymbol{X}_k^i - \boldsymbol{X}_k^j \right\|^2}{2}\right] \end{aligned} \tag{21}$$

where φ is the angle between two estimates, $\boldsymbol{X}_k^i \cdot \boldsymbol{X}_k^j = \sum_{p=1}^{q} X_{k,p}^i X_{k,p}^j$ is inner product of $\boldsymbol{X}_k^i$ and $\boldsymbol{X}_k^j$, q is the dimension of the estimate, and $\| \cdot \|$ is the norm of vector. In fact, (12) not only measures the cosine of angle between two estimates but also the distance between them. As we can see, if $\boldsymbol{X}_k^i = \boldsymbol{X}_k^j$, then $\cos(\varphi) = 1$ and $\exp\left[-\left(\| \boldsymbol{X}_k^i - \boldsymbol{X}_k^j \|^2/2\right)\right] = 1$, which lead to $\mathrm{SIM}\left(\boldsymbol{X}_k^i, \boldsymbol{X}_k^j\right) = 1$; if $\boldsymbol{X}_k^i \perp \boldsymbol{X}_k^j$, then $\cos(\varphi) = 0$, which means $\mathrm{SIM}\left(\boldsymbol{X}_k^i, \boldsymbol{X}_k^j\right) = 0$.

Thus, the similarity matrix for all H estimates is denoted as

$$\boldsymbol{SIMM} = \begin{bmatrix} S_{11} & S_{12} & \cdots & S_{1H} \\ S_{21} & S_{22} & \cdots & S_{2H} \\ \vdots & \vdots & \ddots & \vdots \\ S_{H1} & S_{H2} & \cdots & S_{HH} \end{bmatrix} \tag{22}$$

where $S_{ij} = SIM\left(\boldsymbol{X}_k^i, \boldsymbol{X}_k^j\right)$, $S_{ii} = 1, S_{ij} = S_{ji}$ for $i, j = 1, 2, \ldots, H$. The support degree of estimate is defined as

$$Sup\left(X_k^i\right) = \sum_{i \neq j}^{H} S_{ij}, \quad i, j = 1, 2, \ldots, H \tag{23}$$

It reflects the degree that the estimate is supported by other estimates. If the similarity between one evidence and other evidence is high, the mutual support between them is also high. Also vice versa. The weight of the estimate is defined as

$$w_i = \frac{Sup\left(\boldsymbol{X}_k^i\right)}{\sum_{i=1}^{H} Sup\left(\boldsymbol{X}_k^i\right)}, \quad i = 1, 2, \ldots, H \tag{24}$$

Thus, the fusion state estimate $\hat{\boldsymbol{x}}_k$ and the fusion covariance estimate $\hat{\boldsymbol{P}}_k$ are denoted as

$$\begin{aligned} \hat{\boldsymbol{x}}_k &= \sum_{i=1}^{H} w_i \boldsymbol{X}_k^i, \\ \hat{\boldsymbol{P}}_k &= \sum_{i=1}^{H} w_i \boldsymbol{P}_k^i \end{aligned} \tag{25}$$

In order to better understand the whole process, a general block diagram is presented in Fig. 4. As it can be seen, based on the state prediction by the fusion centre, each node determines to wake up or sleep via the sensor selection scheme. Then, the waking-up nodes estimate the state of the target using ISRCPF and send local estimates to the fusion centre. The fusion centre will fuse them together via SFM to obtain a more precise estimation result and predict the state of the target. At every time, the same procedure is performed recursively.

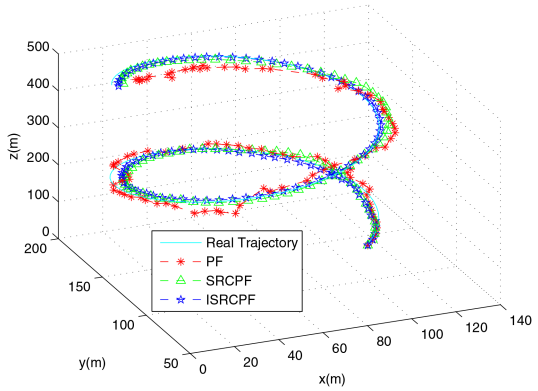

Fig. 5 *Real and estimated trajectories of moving target*

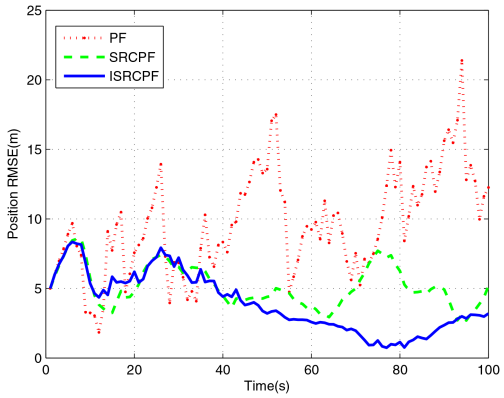

Fig. 6 *Position RMSEs of three filters*

5 Simulation and results

5.1 Simulation setup

The proposed methods are simulated using MATLAB. The considered region is a $600\,\text{m} \times 600\,\text{m} \times 600\,\text{m}$ space with 50 identical sensor nodes deployed randomly. The maximum transmission range and measurement covariance of each node are 300 m and $10\,\text{m}^2$, respectively. Sample time T is assumed one second and time duration is 100 s. To proof the validity of the methods, respectively, we design two experimental scenarios: scenario A and scenario B.

In scenario A, the target makes a constant turn (CT) with turn rate 0.1 rad/s. PF, SRCPF, and ISRCPF are employed to compare the tracking performance. The actual initial state of the target is $\boldsymbol{X}_0^{CT} = [100, 4, 100, 4, 100, 4]^{\mathrm{T}}$, the estimated initial state of the target is $\hat{\boldsymbol{X}}_0^{CT} = [100, 4, 100, 4, 95, 3]$. The initial error covariance $\boldsymbol{P}_0^{(i)} = \boldsymbol{I}_{6\times 6}, i = 1, 2, \ldots, N$. The parameters for SAFSA are set as follows: $s = 1, \alpha = 0.99, D = 30, v = 20, \delta = 0.5$. The number of particles which is used for three tracking algorithms is $N = 500$. In scenario B, it moves at constant velocity (CV). We compare the performance of SFM with local sensors using ISRCPF tracking algorithm. The initial state of the target is $\boldsymbol{X}_0^{CV} = \hat{\boldsymbol{X}}_0^{CV} = [10, 4, 10, 4, 100, 4]^{\mathrm{T}}$. The other parameter settings are the same as those in Scenario A. CT and CV can be formulated as

$$\boldsymbol{X}_k = \boldsymbol{F}_{CT}\boldsymbol{X}_{k-1} + \boldsymbol{w}_k \tag{26}$$

$$\boldsymbol{X}_k = \boldsymbol{F}_{CV}\boldsymbol{X}_{k-1} + \boldsymbol{w}_k \tag{27}$$

where $\boldsymbol{F}_{CT}$ and $\boldsymbol{F}_{CV}$ are state transition matrices, $\boldsymbol{w}_k$ is the process noise with zero-mean, white, Gaussian probability distribution $N(0, \boldsymbol{Q}_k)$. $\boldsymbol{F}_{CT}$, $\boldsymbol{F}_{CV}$ and $\boldsymbol{Q}_k$ are denoted as

$$\boldsymbol{F}_{CT} = \begin{bmatrix} 1 & \frac{\sin(\omega T)}{\omega} & 0 & \frac{\cos(\omega T)-1}{\omega} & 0 & 0 \\ 0 & \cos(\omega T) & 0 & -\sin(\omega T) & 0 & 0 \\ 0 & \frac{1-\cos(\omega T)}{\omega} & 1 & \frac{\sin(\omega T)}{\omega} & 0 & 0 \\ 0 & \sin(\omega T) & 0 & \cos(\omega T) & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & T \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{28}$$

$$\boldsymbol{F}_{CV} = \begin{bmatrix} 1 & T & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & T & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & T \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{29}$$

$$\boldsymbol{Q}_k = q^2 \begin{bmatrix} T^3/3 & T^2/2 & 0 & 0 & 0 & 0 \\ T^2/2 & T & 0 & 0 & 0 & 0 \\ 0 & 0 & T^3/3 & T^2/2 & 0 & 0 \\ 0 & 0 & T^2/2 & T & 0 & 0 \\ 0 & 0 & 0 & 0 & T^3/3 & T^2/2 \\ 0 & 0 & 0 & 0 & T^2/2 & T \end{bmatrix} \tag{30}$$

where ω is turn rate and $q = 0.01\,\text{m/s}^2$ is the intensity of the process noise.

Position RMSE

$$= \sqrt{\frac{1}{R}\sum_{i=1}^{R}\left[\left(X_i(k) - \hat{X}_i(k)\right)^2 + \left(Y_i(k) - \hat{Y}_i(k)\right)^2 + \left(Z_i(k) - \hat{Z}_i(k)\right)^2\right]} \tag{31}$$

Velocity RMSE

$$= \sqrt{\frac{1}{R}\sum_{i=1}^{R}\left[\left(\dot{X}_i(k) - \hat{\dot{X}}_i(k)\right)^2 + \left(\dot{Y}_i(k) - \hat{\dot{Y}}_i(k)\right)^2 + \left(\dot{Z}_i(k) - \hat{\dot{Z}}_i(k)\right)^2\right]} \tag{32}$$

5.2 Simulation metrics

To indicate the accuracy of target tracking, we adopt root mean square error (RMSE) to measure the tracking performance. Considering R Monte Carlo runs, $R = 50$, position RMSE, and velocity RMSE are given by (31) and (32) with respect. Where $(\boldsymbol{X}_i(k), \boldsymbol{Y}_i(k), \boldsymbol{Z}_i(k))$ and $\left(\hat{\boldsymbol{X}}_i(k), \hat{\boldsymbol{Y}}_i(k), \hat{\boldsymbol{Z}}_i(k)\right)$ are the actual location and the estimated location of the target at time k, respectively, and $\dot{\boldsymbol{X}}_i(k), \dot{\boldsymbol{Y}}_i(k), \dot{\boldsymbol{Z}}_i(k)$ denote actual velocity components, while $\hat{\dot{\boldsymbol{X}}}_i(k), \hat{\dot{\boldsymbol{Y}}}_i(k), \hat{\dot{\boldsymbol{Z}}}_i(k)$ denote the estimated velocity components.

5.3 Simulation results

5.3.1 Scenario A: Fig. 5 shows the real and estimated trajectories obtained by PF, SRCPF and ISRCPF. As it can be seen, the target's location is estimated and tracked along the real trajectory. Figs. 6 and 7 show the corresponding position estimation error and the corresponding velocity estimation error with respect. The comparison results suggest that SRCPF performs better than PF, and ISRCPF performs better than SRCPF. According to Table 1, the average position error by the PF is 8.15, by the SRCPF is 5.10, and by the ISRCPF is 2.51; the average velocity error by the PF is 0.94, by the SRCPF is 0.82, and by the ISRCPF is 0.23. These results show significant reductions of estimated position and estimated velocity by the ISRCPF.

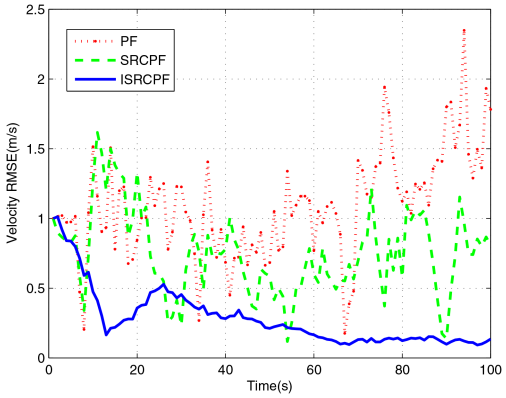

Fig. 7 *Velocity RMSEs of three filters*

Table 1 Comparison of the estimated error for three algorithms

Algorithm	PF	SRCPF	ISRCPF
mean of position RMSE, m	8.15	5.10	**2.51**
mean of velocity RMSE, m/s	0.94	0.82	**0.23**

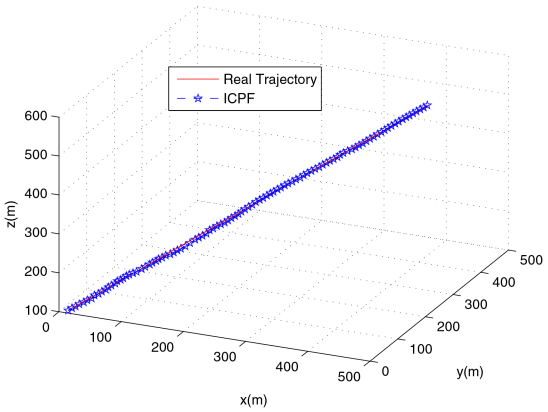

Fig. 8 *Real and estimated trajectories of moving target*

Table 2 Comparison of the estimated error for SFM and local sensors

	Sensor 1	Sensor 2	Sensor 3	Sensor 4	SFM
mean of position RMSE, m	2.97	2.99	2.57	2.74	**2.01**
mean of velocity RMSE, m/s	0.18	0.17	0.16	0.17	**0.12**

The reason for the improvements is that the SRCKF is applied by SRCPF and ISRCPF as the proposal distribution and the use of the current measurement. While the ISRCPF algorithm not only incorporates the current measurement to generate a better importance distribution but also employs SAFSA to optimise the particles driving the particles move towards to high likelihood region. It solves the problem that only the minority of the particles are in the high likelihood area, and thus maintaining the diversity of the particles. Hence, both the phenomenon of particle degeneracy and the problem of particle impoverishment are solved at the same time. Therefore, the location error and the velocity error are converged in the figures. Compared with PF, ISRCPF improves accuracy of the estimated location by 69% better than SRCPF's 37% and improves accuracy of the estimated velocity by 73% better than SRCPF's 14%. ISRCPF may have a small increase in computational time because of using SAFSA. While it is a worthwhile trade in the pursuit of tracking accuracy.

5.3.2 Scenario B: Fig. 8 shows that ISRCPF have successfully completed the mission of target tracking. According to Table 2, it is obvious that estimated location error and estimated velocity error are reduced by SFM. Accuracy of the estimated location error and the estimated velocity error by SFM are improved by 27 and 22%, respectively, comparing to average error of local sensors. The reason for this improvement is that SFM considers the degree of correlation between local estimates. Each estimate is treated as a piece of evidence to determine their confidence by measuring the degree of association between them, thus assigning weight to them. If there is a high degree of similarity between some estimate and other estimate, the mutual support degree of them will be high. The higher the support degree of the estimate, the greater the weight. It suggests that the estimate is closer to real state and naturally and reasonably have a greater influence on the fusion estimate. Thus, the fusion estimate is better than each single local sensor's estimate.

6 Conclusion

Here, an ISRCPF called ISRCPF is proposed for tracking a moving target in 3D underwater environment via UWSNs. The ISRCPF algorithm incorporates the current measurement to generate the proposal distribution and then implements self-adaptive AFSA to optimise the particles. In addition, the sensor selection scheme is utilised. It reduces energy consumption, while preserving tracking accuracy by dynamically exactly selecting four sensor nodes which are closest to predicted position of the target at each time to participate in tracking. Moreover, in distributed fusion architecture, in order to obtain a better estimation result, we propose a novel fusion method named SFM based on the similarity between local estimates. Simulation results demonstrate that the ISRCPF algorithm significantly outperforms the PF and SRCPF algorithms, and the fusion estimate by SFM is superior to each single local sensor's estimate. Therefore, the proposed methods are effective and feasible for underwater target tracking.

7 References

[1] Hui, M., Brian, W.H.N.: 'Collaborative data and information processing for target tracking in wireless sensor networks'. IEEE Int. Conf. on Industrial Informatics, Singapore, Singapore, August 2006, pp. 647–652

[2] Pan, X., Fengju, K., Shengjie, W.: 'Research for underwater target tracking by using multi-sonar'. Int. Congress on Image and Signal Processing, Yantai, China, October 2010, pp. 4249–4253

[3] Poostpasand, M., Javidan, R.: 'An adaptive target tracking method for 3D underwater wireless sensor networks', *Wirel. Netw.*, 2018, **24**, (8), pp. 2797–2810

[4] Zhong, Z., Zheng, P., Junhong, C., *et al.*: 'Scalable localization with mobility prediction for underwater sensor networks', *IEEE Trans. Mob. Comput.*, 2011, **10**, (3), pp. 335–348

[5] Kavoosi, V., Dehghani, M.J., Javidan, R.: 'Selective geometry for near-field three-dimensional localisation using one-pair sensor', *IET Radar Sonar Navig.*, 2016, **10**, (5), pp. 844–849

[6] Javidan, R., Masnadi-Shirazi, M.A, Azimifar, Z.: 'Contourlet-based acoustic seabed ground discrimination system'. Int. Conf. on Information and Communication Technologies: From Theory to Applications, Damascus, Syria, April 2008, pp. 1–6

[7] Dehnavi, S.M., Ayati, M., Zakerzadeh, M.R.: 'Three dimensional target tracking via underwater acoustic wireless sensor network'. Artificial Intelligence and Robotics, Qazvin, Iran, April 2017, pp. 153–157

[8] Isbitiren, G., Akan, O.B.: 'Three-dimensional underwater target tracking with acoustic sensor networks', *IEEE Trans. Veh. Technol.*, 2011, **60**, (8), pp. 3897–3906

[9] Wei, L., Yiping, L., Shenzhen, R., *et al.*: 'Tracking an underwater manoeuvring target using an adaptive Kalman filter'. Tencon IEEE Region 10 Conf., Xi'an, China, October 2013, pp. 1–4

[10] Dengfeng, M., Kzaizhou, L., Yanyan, W.: 'MEFPDA-SCKF for underwater single observer bearings-only target tracking in clutter'. OCEANS, San Diego, USA, September 2014, pp. 1–6

[11] Munjani, J.H., Joshi, M.: 'Target tracking in WSN using time delay neural network'. IEEE Region 10 Conf., Singapore, Singapore, November 2016, pp. 3839–3845

[12] Guangzhu, C., Lijuan, Z., Zhencai, Z., *et al.*: 'RBF neural network based prediction for target tracking in chain-type wireless sensor networks'. The 2nd IEEE Int. Conf. on Advanced Computer Control, Shenyang, China, March 2010, pp. 635–639

[13] Arulampalam, M.S, Maskell, S., Gordon, N., *et al.*: 'A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking', *IEEE Trans. Signal Process.*, 2002, **50**, (2), pp. 174–188

[14] Linlin, Z., Rijie, Y., Hua, X.: 'Application of non-linear filtering to underwater target tracking', *Fire Control Command Control*, 2010, **35**, (8), pp. 13–17

[15] Yan, H., Wei, L., Haibin, Y., *et al.*: 'Target tracking based on a distributed particle filter in underwater sensor networks', *Wirel. Commun. Mob. Comput.*, 2008, **8**, (8), pp. 1023–1033

[16] Jing, M., Yanli, C., Junmin, Z.: 'Square root cubature particle filter', *Adv. Mater. Res.*, 2011, **219**, (1), pp. 727–731
[17] Karlsson, R., Bergman, N.: 'Auxiliary particle filters for tracking a manoeuvring target'. IEEE Conf. on Decision & Control, Sydney, Australia, December 2002, pp. 3891–3895
[18] Supeng, C., Weimin, H.: 'Manoeuvring target tracking from nautical radar images using particle-Kalman filters', *J. Electromagn. Waves Appl.*, 2013, **27**, (18), pp. 2366–2378
[19] Havangi, R.: 'Target tracking based on improved unscented particle filter with Markov chain Monte Carlo', *IETE J. Res.*, 2018, **64**, (6), pp. 873–885
[20] Huayan, C., Senlin, Z., Meiqin, L.*, et al.*: 'An artificial measurements-based adaptive filter for energy-efficient target tracking via underwater wireless sensor networks', *Sensors*, 2017, **17**, (5), pp. 1–19
[21] Ashkooti, F., Rashidy, R.: 'A distributed particle filter for acustic target tracking in wireless sensor networks'. IEEE Int. Conf. on Application of Information and Communication Technologies, Baku, Azerbaijan, October 2017, pp. 1–6
[22] Qiang, Z., Chaojie, Z., Meiqin, L.*, et al.*: 'Local node selection for target tracking based on underwater wireless sensor networks', *Int. J. Syst. Sci.*, 2015, **46**, (16), pp. 2918–2927
[23] Hare, J., Gupta, S., Song, J.: 'Distributed smart sensor scheduling for underwater target tracking'. OCEANS, St. John's, Canada, September 2014, pp. 1–6
[24] Watfa, M.K., Nsouli, T., Ayach, M.E.*, et al.*: 'Reactive localisation in underwater wireless sensor networks with self-healing', *Int. J. Intell. Syst. Technol. Appl.*, 2013, **12**, (1), pp. 63–85
[25] Ying, Z., Jixing, L., Shengming, J.*, et al.*: 'A localization method for underwater wireless sensor networks based on mobility prediction and particle swarm optimization algorithms', *Sensors*, 2016, **16**, (2), pp. 1–17
[26] Xiaolei, L., Zhijiang, S., Jixin, Q.: 'An optimizing method based on autonomous animats: fish-swarm algorithm', *Syst. Eng. Theory Pract.*, 2002, **22**, (11), pp. 32–38
[27] Jiye, L., Xihong, C., Qiang, L.*, et al.*: 'Prediction of satellite clock errors using LS-SVM optimized by improved artificial fish swarm algorithm'. IEEE Int. Conf. on Signal Processing, Communication and Computing, KunMing, China, August 2013, pp. 1–5