Journal of Systems Engineering and Electronics
Vol. 33, No. 4, August 2022, pp.867 – 876

Improved IMM algorithm based on support vector regression for UAV tracking

ZENG Yuan[1,2], LU Wenbin[1], YU Bo[3], TAO Shifei[3], ZHOU Haosu[1], and CHEN Yu[2,*]

1. Shanghai Spaceflight Electronic and Communication Equipment Research Institute, Shanghai 201109, China; 2. Science and Technology on Near-Surface Detection Laboratory, Wuxi 214035, China; 3. School of Electronic and Optical Engineering, Nanjing University of Science and Technology, Nanjing 210094, China

Abstract: With the development of technology, the relevant performance of unmanned aerial vehicles (UAVs) has been greatly improved, and various highly maneuverable UAVs have been developed, which puts forward higher requirements on target tracking technology. Strong maneuvering refers to relatively instantaneous and dramatic changes in target acceleration or movement patterns, as well as continuous changes in speed, angle, and acceleration. However, the traditional UAV tracking algorithm model has poor adaptability and large amount of calculation. This paper applies support vector regression (SVR) to the interacting multiple model (IMM) algorithm. The simulation results show that the improved algorithm has higher tracking accuracy for highly maneuverable targets than the original algorithm, and can adjust parameters adaptively, making it more adaptable.

Keywords: interacting multiple model (IMM) filter, constant acceleration (CA), unmanned aerial vehicle (UAV), support vector regression (SVR).

DOI: 10.23919/JSEE.2022.000075

1. Introduction

Maneuvering targets are the current difficulty in target detection and tracking. Due to their strong maneuverability, it is difficult for the single-model method to track them with high precision and it is easy to lose targets. The interactive multi-model (IMM) method proposed by Blom et al. [1] is one of the methods widely used in maneuvering target tracking algorithms. This method simulates the movement of the targets by establishing a set of models and assumes that the switching of different models follows the Markov process, and the final state of the target is finally weighted and determined by the filter values of different models [2].

The IMM algorithm uses multiple motion models in parallel and uses a Markov process to simulate the transition between models, and finally weights the filtered values of multiple models as an estimator of the target state. For maneuvering targets, the algorithm’s tracking effect is better than that of the single model algorithm. Its adaptability is also better. Compared with the single-model algorithm, the IMM algorithm has many advantages [3]:

(i) The IMM algorithm contains multiple models, and the number and types of models in the model set can be set according to actual needs.

(ii) In the process of target tracking, the accuracy of target tracking can be improved by adjusting the probability of the model.

(iii) Each model has its own filter, and the filters of each model can be adjusted according to actual needs to improve the filtering performance.

However, the IMM algorithm has some shortcomings. First of all, similar to the early multi-model algorithm and some other prior models, the performance of the interactive multi-model algorithm depends to a large extent on the model used. Considering the amount of computation and the competition between models, the size of the model set should not be set too large, and it should be set in advance because different models have different processing methods. Moreover, too many models in the model set will lead to competition between the models, thereby reducing the accuracy of the algorithm [4]. Therefore, a model set of proper scale should be established in advance. Once the model set is determined, the model set will not be changed in the tracking process. However, with the development of control technology, the mobility of various targets is getting better and better, and the preset fixed number of model sets is difficult to meet actual needs.

To solve the problems caused by the fixed model set,

Manuscript received December 23, 2020.
*Corresponding author.
This work was supported by the Foundation of Key Laboratory of Near-Surface.

Li et al. proposed a variable structure multi model algorithm in 1999 [5], and then proposed several practical algorithms based on variable structure multiple models: possible model set algorithm, maximum likelihood function algorithm, and grid algorithm [6] in 2019. Based on the above methods, most of the variable structure multi-model algorithms have been researched [7–9]. Many experts and scholars have also proposed improved algorithms to solve the problem of low tracking accuracy of maneuvering targets due to poor adaptive filter. In [10], a fuzzy adaptive controller was designed to join the nonlinear system to adjust the adaptiveness of parameters. In [11], the fuzzy membership function was introduced into the current statistical model to achieve the adaptive adjustment of the target acceleration. In [12], a fuzzy inference system was introduced into the tracking algorithm to reduce the tracking error by adaptively adjusting the maneuvering frequency.

To solve the problem of poor IMM adaptability, this paper applies support vector regression (SVR) to the IMM algorithm. The IMM algorithm using SVR is tested through simulation. The improved IMM algorithm not only has low model complexity, but also improves the tracking accuracy of maneuvering targets.

This paper is organized as follows. Section 2 mainly introduces the detailed principles and steps of the IMM algorithm, the SVR algorithm, and the improved IMM algorithm. Section 3 mainly introduces the IMM algorithm and the improved IMM algorithm simulation results. Section 4 presents conclusions.

2. Methodology

2.1 IMM algorithm

The basic idea of the multi-model (MM) algorithm is to give a model set containing one or more motion models (including non-motion model and motion model) before the state estimation, and then use multiple models to filter and estimate separately in parallel. Finally, the state filter values of each model are weighted and summed according to the probabilities of each model in the current period [13].

The IMM algorithm uses multiple different motion models to match the different motion states of the target. The transition probability between different models is a Markov chain [14], and the target state is estimated using a Kalman filter. It mainly has the following components:

Step 1 Input interaction module.

Input the target state filter value and covariance filter value of each model in the previous cycle period into the input interaction module and perform input interaction according to the model probability transition matrix and the model probability. Then put interaction values into each model [15].

$$\hat{\boldsymbol{X}}^{\mathrm{o}j}(k-1|k-1)=\sum_{i=1}^{N}\hat{\boldsymbol{X}}^{i}(k-1|k-1)\cdot u_{k-1|k-1}(i|j) \tag{1}$$

In (1), $\hat{\boldsymbol{X}}^{\mathrm{o}i}(k-1|k-1)$ is the input of the ith filter at time k after interactive calculation, $\boldsymbol{P}^{i}(k-1|k-1)$ is the corresponding state covariance. $\hat{\boldsymbol{X}}^{i}(k-1|k-1)$ is the state estimation of the ith filter at k−1, and $\boldsymbol{P}^{\mathrm{o}j}(k-1|k-1)$ is the corresponding state covariance. $u_{k-1|k-1}(i|j)$ is the transition probability of the ith filter during interactive calculation.

$$\begin{aligned}\boldsymbol{P}^{\mathrm{o}j}(k-1|k-1)=&\sum_{i=1}^{N}\Big\{\boldsymbol{P}^{i}(k-1|k-1)+\\&\left[\hat{\boldsymbol{X}}^{i}(k-1|k-1)-\hat{\boldsymbol{X}}^{\mathrm{o}j}(k-1|k-1)\right]\cdot\\&\left[\hat{\boldsymbol{X}}^{i}(k-1|k-1)-\hat{\boldsymbol{X}}^{\mathrm{o}j}(k-1|k-1)\right]^{\mathrm{T}}\Big\}\cdot u_{k-1|k-1}(i|j)\end{aligned} \tag{2}$$

Step 2 Model filter estimation module.

Each model uses its own filter to filter the input state interaction value and input covariance interaction value to obtain the state estimate and covariance estimate. $\hat{\boldsymbol{X}}^{i}(k|k)$ is the state estimate of the ith filter at time k. $\boldsymbol{K}(k-1)$ is the filter gain at k−1. $\boldsymbol{V}(k-1)$ is the innovation sequence at time k−1.

$$\hat{\boldsymbol{X}}^{i}(k|k)=\hat{\boldsymbol{X}}^{i}(k|k-1)+\boldsymbol{K}(k-1)\boldsymbol{V}(k-1) \tag{3}$$

Step 3 Update of model probability module.

Update the model probability value according to the estimated value of each model and the measurement value received by the sensor at the current moment. $\wedge_k^j$ is the probability of the jth model, $\boldsymbol{v}_k^j$ is the filter residual, $\boldsymbol{S}_k^j$ is the corresponding covariance. The probability of the jth model is updated to

$$\wedge_k^j=\frac{1}{\sqrt{|2\pi\boldsymbol{S}_k^j|}}\exp\left[-\frac{1}{2}\left(\boldsymbol{v}_k^j\right)'\left(\boldsymbol{S}_k^j\right)^{-1}\boldsymbol{v}_k^j\right], \tag{4}$$

$$u_k(j)=\frac{1}{C}\wedge_k^j\overline{C}_j, \tag{5}$$

$$C=\sum_{i=1}^{N}\wedge_k^i\overline{C}_i. \tag{6}$$

Step 4 Estimation fusion module.

IMM is an algorithm with data fusion as the core. Probability weighted summation of the estimated value of each model is used as the target state estimation for each cycle [16]. $\hat{\boldsymbol{X}}(k|k)$ and $\boldsymbol{P}(k|k)$ are the interactive output at time k respectively.

$$\hat{X}(k|k)=\sum_{i=1}^{N}\hat{X}^{i}(k|k)u_k(i) \tag{7}$$

$$P(k|k)=\sum_{i=1}^{N}u_k(i)\left\{P^i(k|k)+\left[\hat{X}^{i}(k|k)-\hat{X}(k|k)\right]\left[\hat{X}^{i}(k|k)-\hat{X}(k|k)\right]^{\mathrm{T}}\right. \tag{8}$$

A filtering cycle of the IMM algorithm mainly includes four steps: input interaction module, model filter estimation module, update of model probability module, estimation fusion module. The algorithm block diagram is shown in Fig. 1.

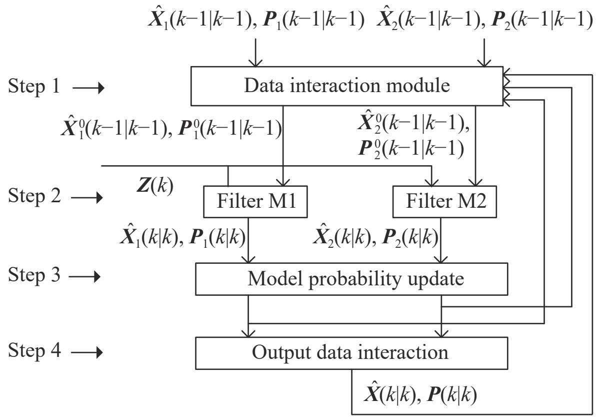

Fig. 1 IMM algorithm flow chart

2.2 SVR

Support vector machine (SVM) originally came from classification and then is used to solve regression problems, called ε-SVR of SVR machine model [17].

In regression problems, we give a training data set. The data set consists of input and label. There are n sets of data which is $G_0=\{(\boldsymbol{x}_i,y_i):\boldsymbol{x}_i=(x_i^1,x_i^2,\cdots,x_i^d)^{\mathrm{T}},\ y_i\in\mathbf{R}\}_{i=1}^n$ in d-dimensional space. Constructing a regression function $f(\boldsymbol{x}_i)=\boldsymbol{w}^{\mathrm{T}}\cdot\varphi(\boldsymbol{x}_i)+b$, where $\varphi(\boldsymbol{x}_i)$ is the nonlinear mapping function, $b\in\mathbf{R}$ is the threshold, and $\boldsymbol{w}$ is the feature weight vector.

Step 1 Set-up of the convex quadratic optimization problem.

Quote the linear insensitive loss function:

$$L(f(x),y,\theta)=\begin{cases}0, & |y-f(x)|\leqslant\varepsilon\\ |y-f(x)|-\varepsilon, & |y-f(x)|>\varepsilon\end{cases} \tag{9}$$

where $f(x)$ is the predicted value of the fitting function, y is the predicted value of the fitting function. Disregarding the small errors that fall within some tolerance, say ε, may lead to a better generalization ability achieved by utilizing an ε-insensitive loss function.

The significance of the insensitive loss function reference is that if the difference between $f(x)$ and y is within the allowable error range, then $f(x)$ has no loss.

Introduce slack variable $\boldsymbol{\xi}$ and establish the following constraints:

$$\min_{(\boldsymbol{w},b)\in\mathbf{R}^{n+1}}\frac{1}{2}\boldsymbol{w}^{\mathrm{T}}\boldsymbol{w}+C\mathbf{1}^{\mathrm{T}}|\boldsymbol{\xi}|_i \tag{10}$$

where $\mathbf{1}$ denotes the $m\times1$ all-one vector, $|\boldsymbol{\xi}|_i\in\mathbf{R}^m$, $(|\boldsymbol{\xi}|)_i=\max\{0,|\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{w}+b-y_i|-\varepsilon\}$ that represent the fitting errors and the positive control parameter C here weights the tradeoff between the fitting errors and the flatness of the linear regression function $f(x)$. Rewrite the above formula into the following formula:

$$\min_{(\boldsymbol{w},b,\xi,\xi^*)\in\mathbf{R}^{n+1+2m}}\frac{1}{2}\boldsymbol{w}^{\mathrm{T}}\boldsymbol{w}+C\sum_{i=1}^{m}(\xi_i+\xi_i^*)$$
$$\text{s.t.}\begin{cases}y_i-\boldsymbol{w}\cdot\varphi(\boldsymbol{x}_i)-b\leqslant\varepsilon+\xi_i\\ -y_i+\boldsymbol{w}\cdot\varphi(\boldsymbol{x}_i)+b\leqslant\varepsilon+\xi_i^*\\ \xi_i\geqslant0,\ \xi_i^*\geqslant0,\ i=1,2,\cdots,n\end{cases}. \tag{11}$$

Step 2 Lagrange dual problem.

Introduce Lagrange coefficients μ_i and transform them into dual form:

$$L(\boldsymbol{w},b,\boldsymbol{\alpha},\boldsymbol{\alpha}^*,\boldsymbol{\xi},\boldsymbol{\xi}^*,\boldsymbol{\mu},\boldsymbol{\mu}^*)=\frac{1}{2}\|\boldsymbol{w}\|^2+C\sum_{i=1}^{m}(\xi_i+\xi_i^*)-\sum_{i=1}^{m}\xi_i\mu_i-\sum_{i=1}^{m}\xi_i^*\mu_i^*+\sum_{i=1}^{m}\alpha_i(f(x_i)-y_i-\varepsilon-\xi_i)+\sum_{i=1}^{m}\alpha_i^*(y_i-f(x_i)-\varepsilon-\xi_i^*). \tag{12}$$

Then let the partial derivative of L to $\boldsymbol{w},b,\xi_i$, and ξ_i^* be 0 to obtain

$$\begin{cases}\boldsymbol{w}=\sum_{i=1}^{m}(\alpha_i^*-\alpha_i)\boldsymbol{x}_i\\ \sum_{i=1}^{m}(\alpha_i^*-\alpha_i)=0\\ \alpha_i+\mu_i=C\\ \alpha_i^*+\mu_i^*=C\end{cases}. \tag{13}$$

Take it back to the Lagrange function, simplify it to get a function only about α_{i} and α_{i}^*. The goal is to maximize this function.

$$L(\boldsymbol{\alpha},\boldsymbol{\alpha}^*)=-\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}(\alpha_i-\alpha_i^*)\left(\alpha_j-\alpha_j^*\right)\boldsymbol{K}(x_i,x_j)-\varepsilon\sum_{i=1}^{m}(\alpha_i+\alpha_i^*)+\sum_{i=1}^{m}(\alpha_i+\alpha_i^*)y_i^* \tag{14}$$

The constraints are

$$\begin{cases} 0 \leqslant \alpha_i \leqslant C \\ 0 \leqslant \alpha_i^* \leqslant C \end{cases} \tag{15}$$

where $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i)^{\mathrm{T}} \boldsymbol{x}_j$ is the vector inner product. It is now a linear kernel, and it can also be replaced with a nonlinear kernel function such as a Gaussian kernel.

In the above process, Karush-Kuhn-Tucker (KKT) conditions must be met, that is,

$$\begin{cases} \alpha_i (f(\boldsymbol{x}_i) - y_i - \theta - \xi_i) = 0 \\ \alpha_i^* (y_i - f(\boldsymbol{x}_i) - \theta - \xi_i^*) = 0 \\ \alpha_i \alpha_i^* = 0 \\ \xi_i \xi_i^* = 0 \\ (C - \alpha_i) \xi_i = 0 \\ (C - \alpha_i^*) \xi_i^* = 0 \end{cases} \tag{16}$$

From (16), we know

(i) When $\alpha_{\mathrm{i}} > 0$, there must be

$$\varepsilon + \xi_i + \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}_i + b - y_i^* = 0, \quad \xi_i \geqslant 0. \tag{17}$$

These points are located at the upper boundary of the pipe, or above the pipe. The predicted value is smaller than the true value.

(ii) When $\alpha_i^* > 0$, there must be

$$\varepsilon + \xi_i^* \geqslant -\left(\boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}_i + b\right) + y_i^* = 0, \quad \xi_i^* \geqslant 0. \tag{18}$$

These points are located at the lower boundary of the pipe, or below the pipe. The predicted value is greater than the true value [18].

At the same time, from (16) we know that for any data point, since $\varepsilon > 0$, it is impossible for both a and b to be greater than 0 at the same time, and to get a point inside the pipeline, there must be $\alpha_i = 0, \alpha_i^* = 0$.

Step 3 Hyperplane computing.

According to the previous calculations, we can get

$$\boldsymbol{w} = \sum_{i=1}^{m} (\alpha_i^* - \alpha_i) \boldsymbol{x}_i. \tag{19}$$

From the above analysis, the point that affects the hyperplane parameters is located at the pipe boundary or outside the pipe. Regarding the calculation of b, it can be considered that a point at the upper boundary of the pipeline must have

$$\begin{cases} \xi_i = 0 \\ \varepsilon + \xi_i + \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}_i + b - y_i^* = 0 \end{cases}. \tag{20}$$

It can be solved that

$$\begin{aligned} b = y_i^* - \varepsilon - \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}_i = \\ y_i^* - \varepsilon - \sum_{j=1}^{m} \left(\alpha_j - \alpha_j^*\right) (\boldsymbol{x}_j)^{\mathrm{T}} \boldsymbol{x}_i = \\ y_i^* - \varepsilon - \sum_{j=1}^{m} \left(\alpha_j - \alpha_j^*\right) \boldsymbol{K}(\boldsymbol{x}_i, \boldsymbol{x}_j) \end{aligned} \tag{21}$$

Then the prediction function is

$$\begin{aligned} y(\boldsymbol{x}) = \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x} + b = \\ \sum_{i=1}^{m} (\alpha_i - \alpha_i^*)(\boldsymbol{x}_i)^{\mathrm{T}} \boldsymbol{x} + b = \\ \sum_{i=1}^{m} (\alpha_i - \alpha_i^*) \boldsymbol{K}(\boldsymbol{x}_i, \boldsymbol{x}) + y_j^* - \\ \varepsilon - \sum_{i=1}^{m} (\alpha_i - \alpha_i^*) \boldsymbol{K}(\boldsymbol{x}_i, \boldsymbol{x}_j) \end{aligned} \tag{22}$$

where $\boldsymbol{x}$ is a point on the plane boundary of the hyperplane pipe.

Step 4 Proof of convergence.

In our smooth approach, we change the model slightly and solve it as an unconstrained minimization problem directly without adding any new variable and constraint. That is, the squares of 2-norm ε-insensitive loss. In addition, we add the term $b^2/2$ in the objective function to induce strong convexity and to guarantee that the problem has a unique global optimal solution [19].

Therefore, (11) can also be rewritten as

$$\min_{(\boldsymbol{w}, b) \in \mathbf{R}^{n+1}} \frac{1}{2} (\boldsymbol{w}^{\mathrm{T}} \boldsymbol{w} + b^2) + \frac{C}{2} \sum_{i=1}^{m} |\boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{w} + b - y_i|_{\varepsilon}^2. \tag{23}$$

Inspired by smooth support vector machine (SSVM) for classification, the squares of ε-insensitive loss function in the above formulation can be accurately approximated by a smooth function which is infinitely differentiable and defined below:

$$\begin{aligned} |x|_{\varepsilon} = \max\{0, |x| - \varepsilon\} = \\ (x - \varepsilon)_+ + (-x - \varepsilon)_+. \end{aligned} \tag{24}$$

Furthermore, $(x - \varepsilon)_+ \cdot (-x - \varepsilon)_+ = 0$ for all $x \in \mathbf{R}$ and $\varepsilon > 0$. Thus, we have

$$|x|_{\varepsilon}^2 = (x - \varepsilon)_+^2 + (-x - \varepsilon)_+^2. \tag{25}$$

In SSVM, the plus function x_+ is approximated by a smooth p-function, $p(x, \alpha) = p(x - \varepsilon, \alpha))^2 + (p(-x - \varepsilon, \alpha))^2$, $\alpha > 0$.

Therefore,

$$p_{\varepsilon}^2(x, \alpha) = (p(x - \varepsilon, \alpha))^2 + (p(-x - \varepsilon, \alpha))^2. \tag{26}$$

The original objective function can be expressed as

$$\min_{(\boldsymbol{w}, b) \in \mathbf{R}^{n+1}} \frac{1}{2} (\boldsymbol{w}^{\mathrm{T}} \boldsymbol{w} + b^2) + \frac{C}{2} \mathbf{1}^{\mathrm{T}} p_{\varepsilon}^2 (\boldsymbol{x} \boldsymbol{w} + \mathbf{1} b - \boldsymbol{y}, \alpha). \tag{27}$$

Rewrite (23) and (27) as

$$h_{\varepsilon}(x) = \frac{1}{2} (\boldsymbol{w}^{\mathrm{T}} \boldsymbol{w} + b^2) + \frac{C}{2} \sum_{i=1}^{m} |\boldsymbol{x}_i \boldsymbol{w} + b - y_i|_{\varepsilon}^2, \tag{28}$$

$$g_\varepsilon(x,\alpha)=\frac{1}{2}(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{w}+b^2)+\frac{C}{2}\boldsymbol{1}^{\mathrm{T}}p_\varepsilon^2(\boldsymbol{x}\boldsymbol{w}+\boldsymbol{1}^{\mathrm{T}}b-\boldsymbol{y},\alpha). \quad (29)$$

Combine $\boldsymbol{w}$ and b into a vector $\boldsymbol{t}=(\boldsymbol{w},b)^{\mathrm{T}}$. Mark $\tilde{\boldsymbol{x}}=(\boldsymbol{x}:\boldsymbol{1}^{\mathrm{T}})$. Therefore, the formula is rewritten as

$$h_\varepsilon(\boldsymbol{x})=\frac{1}{2}\|\boldsymbol{t}\|_2^2+\frac{C}{2}\sum_{i=1}^{m}|\tilde{\boldsymbol{x}}_i\boldsymbol{t}-y_i|_\varepsilon^2, \quad (30)$$

$$g_\varepsilon(\boldsymbol{x},\alpha)=\frac{1}{2}\|\boldsymbol{t}\|_2^2+\frac{C}{2}\sum_{i=1}^{m}p_\varepsilon^2(\tilde{\boldsymbol{x}}_i\boldsymbol{t}-y_i,\alpha). \quad (31)$$

To prove the convergence, three lemmas and a theorem are introduced as follows:

(i) $|\boldsymbol{x}|_\varepsilon^2$ and $p_\varepsilon^2(\boldsymbol{x},\alpha)$ are given by (25) and (26), therefore,

$$p_\varepsilon^2(\boldsymbol{x},\alpha)>|\boldsymbol{x}|_\varepsilon^2. \quad (32)$$

(ii) The objective function is defined as shown in (30) and (31), then the optimization problems $\min\limits_{\boldsymbol{x}} h_\varepsilon(\boldsymbol{x})$ and $\min\limits_{\boldsymbol{x}} g_\varepsilon(\boldsymbol{x},\alpha)$ have solutions.

(iii) Assume that the solutions of the optimization problems $\min\limits_{\boldsymbol{x}} h_\varepsilon(\boldsymbol{x})$ and $\min\limits_{\boldsymbol{x}} g_\varepsilon(\boldsymbol{x},\alpha)$ are $\bar{\boldsymbol{x}}$ and $\bar{\boldsymbol{x}}_\alpha$, therefore,

$$h_\varepsilon(\boldsymbol{x})-h_\varepsilon(\bar{\boldsymbol{x}})\geqslant\frac{1}{2}\|\boldsymbol{x}-\bar{\boldsymbol{x}}\|_2^2, \quad (33)$$

$$g_\varepsilon(\boldsymbol{x},\alpha)-g_\varepsilon(\bar{\boldsymbol{x}},\alpha)\geqslant\frac{1}{2}\|\boldsymbol{x}-\bar{\boldsymbol{x}}_\alpha\|_2^2. \quad (34)$$

(iv) For any $C>0$, the solution of the optimization problems $\min\limits_{\boldsymbol{x}} g_\varepsilon(\boldsymbol{x},\alpha)$ are globally convergent to the solution of $\min\limits_{\boldsymbol{x}} h_\varepsilon(\boldsymbol{x})$.

$$h(\bar{\boldsymbol{x}}_\alpha)-h(\bar{\boldsymbol{x}})\geqslant\frac{1}{2}\|\boldsymbol{x}-\bar{\boldsymbol{x}}\|_2^2 \quad (35)$$

$$g(\bar{\boldsymbol{x}}_\alpha,\alpha)-g(\bar{\boldsymbol{x}}_\alpha)\geqslant\frac{1}{2}\|\bar{\boldsymbol{x}}-\bar{\boldsymbol{x}}_\alpha\|_2^2 \quad (36)$$

Formula (35) plus (36) gets

$$\|\bar{\boldsymbol{x}}_\alpha-\bar{\boldsymbol{x}}\|_2^2\leqslant(g_\varepsilon(\bar{\boldsymbol{x}},\alpha)-h_\varepsilon(\bar{\boldsymbol{x}}))-(g_\varepsilon(\bar{\boldsymbol{x}},\alpha)-h_\varepsilon(\bar{\boldsymbol{x}}_\alpha)). \quad (37)$$

Combine the three lemmas to get

$$\|\bar{\boldsymbol{x}}_\alpha-\bar{\boldsymbol{x}}\|_2^2\leqslant g_\varepsilon(\bar{\boldsymbol{x}},\alpha)-h_\varepsilon(\bar{\boldsymbol{x}})=\left(\frac{C}{2}\sum_{\mathrm{i}=1}^{m}p_\varepsilon^2(\tilde{\boldsymbol{x}}_i\bar{\boldsymbol{x}}-y_i,\alpha)-|\tilde{\boldsymbol{x}}_i\bar{\boldsymbol{x}}_i-y_i|_\varepsilon^2\right). \quad (38)$$

Convergence has been proved as

$$p_\varepsilon^2(\boldsymbol{x},\alpha)-|\boldsymbol{x}|_\varepsilon^2\leqslant2\left(\frac{\ln2}{\alpha}\right)^2+\frac{2\rho}{\alpha}\ln2 \quad (39)$$

where ρ is a constant.

Bring (39) into (38) to get

$$\|\bar{\boldsymbol{x}}_\alpha-\bar{\boldsymbol{x}}\|_2^2\leqslant mC\left[\left(\frac{\ln2}{\alpha}\right)^2+\frac{\rho}{\alpha}\ln2\right]. \quad (40)$$

It is proved that the support vector regression machine is globally convergent.

2.3 Improved IMM algorithm

Finally, an improved IMM algorithm is introduced, which is improved based on the IMM algorithm [20]. As shown in Fig. 2, on the basis of the original IMM algorithm, an SVR machine network is added as a feedback network. The model set is still preset, but the feedback network will monitor the matching degree of different models with the actual movement of the target in real time. Then, adjust the process noise covariance matrix coefficients of different model filters according to the degree of matching. Finally, the tracking error can be reduced when the model does not match.

Based on the above discussion, we should first determine the matching degree of the model in real time. The simulation shows that in target tracking, when the target is maneuvering, the filtering residual becomes larger and the greater the maneuvering intensity, the larger the filtering residual [21]. Therefore, the filtered residual information is an important parameter for testing the strength of the target's maneuverability, and it is also an important parameter for testing the degree of model matching. The following formula is now used as the real-time monitoring of the network.

$$\bar{\boldsymbol{d}}(k)=z(k)-\hat{z}(k)=\begin{pmatrix}\sqrt{\dfrac{\Delta x}{\sigma_x}}\\ \sqrt{\dfrac{\Delta y}{\sigma_y}}\end{pmatrix} \quad (41)$$

In addition, simulation shows that when the model does not match, reducing the process noise covariance coefficient can reduce the overall tracking error. Therefore, different matching degrees must correspond to a coefficient. Taking innovation as the input of the network, the optimal coefficient as the output value, supplemented by a large amount of training data, the feedback network can be obtained.

The module of coefficient adjustment based on SVR is responsible for receiving the relevant parameters of the IMM model, thus output the coefficient of the error covariance matrix (ECM), and then return this parameter to the IMM model for estimating the current state of the target. The coefficient about ECM is actually used as an intermediate coefficient of the overall tracking model. Therefore, the real state of the target at the current moment can be used as the label, and the loss function for the training of the constraint network is set as the mean square error (MSE) between the estimated state of the target based on the ECM and the real state. The formula for the loss function is as follows:

$$\text{loss} = \sqrt{\frac{1}{m}\sum_{i=1}^{m}\left(h\left(a_{\max}^{i}\right)-y^{(i)}\right)^{2}}. \tag{42}$$

In the simulation conditions, there are cases of model matching and mismatch, with different adjusting coefficients. Adjusting coefficient and model training are constrained by the above loss function. Thus as long as Monte Carlo simulation is carried out for several times, the corresponding training set and label can be generated. The model training set is the innovation corresponding to different models for several times, and the label is the real state of the target at the current moment.

Since the simulation scenario and model set only involves two simple models, the size of adjustment coefficients is only two. Therefore, the function of adjustment about ECM (SVR module) is not a very complex function. Therefore, a limited training set can fully train a well-performing SVR module. In addition, the training and test scenarios are separate. In the training scenario, the time series involved in motion is 400, which contains scenarios with different model matching degrees. After several Monte Carlo simulations, 1 000 sets of data are collected as the training set.

Now suppose that two models are set up in the model set, then the improved IMM algorithm schematic diagram is shown in Fig. 2.

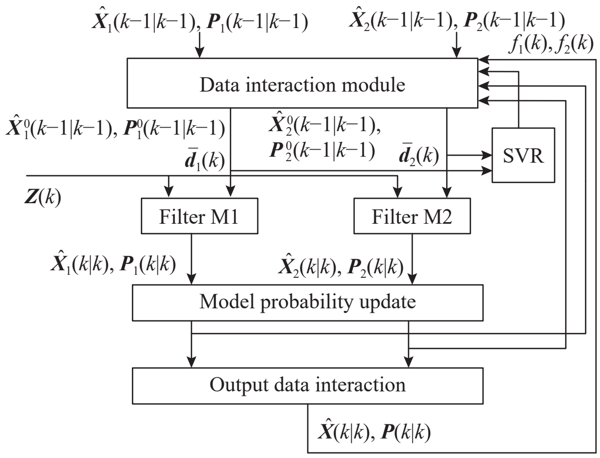

Fig. 2 Improved IMM algorithm flow chart

It mainly has the following components.

Step 1 Input the filtered value of each model in the previous cycle to the input interaction module for input interaction, and then input the interaction value of each model into each model for a new round of filtering.

Step 2 Each model performs filtering according to the measurement of the current period, inputs interaction value, and outputs the state estimate and estimate of covariance.

Step 3 Each model updates the model probability value according to the relevant result parameters.

Step 4 The prediction error $d_1(k)$ and $d_2(k)$ of two models are used as the input of SVR, and the coefficients $f_1(k)$ and $f_2(k)$ of the covariance matrix are output, which can feed back to Step 2 to adjust the estimation results.

Step 5 Calculate the overall state output using the current target state estimates of each filter and the probability of each model.

The specific steps of the improved IMM algorithm are basically the same as those of the standard IMM algorithm, which are also divided into: (i) filter initialization and data interaction; (ii) model filter estimation; (iii) model probability update; (iv) data estimation fusion. Among them, Step 1, Step 3, and Step 4 are the same as the original algorithm, and the covariance part prediction of the original algorithm in Step 2 is replaced by

$$\overline{\boldsymbol{P}}^{(i)}(k|k-1) = \boldsymbol{F}^{(i)}(k-1)\cdot$$
$$\bar{\boldsymbol{P}}^{(i)}(k-1|k-1)\left(\boldsymbol{F}^{(i)}(k-1)\right)^{\mathrm{T}} + f^{(i)}(k-1)\boldsymbol{Q}^{(i)}(k-1) \tag{43}$$

where $f^{(i)}(k-1)$ represents the output component of the SVR system, and its initial value is 1.

3. Experiments

Assume that the target moves in two-dimensional space, the sampling period T=1 s, the measurement noise is a Gaussian sequence with a mean value of 0 and a standard deviation of 100 m, and the number of simulations is 200. If the IMM is composed of two single models, one is the constant velocity model, the other is the constant turning model. The initial probabilities of the models are 1/2, and the model transition probability matrix is

$$\boldsymbol{A} = \begin{bmatrix} 0.9 & 0.1 \\ 0.1 & 0.9 \end{bmatrix}.$$

A total of 200 Monte-Carlo simulations are used, and the evaluation index is the target root mean square error (RMSE) of position and speed, which can be represented as

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{j=1}^{N}\left(\boldsymbol{X}(k)-\overset{j}{\hat{\boldsymbol{X}}}(k|k)\right)^{2}} \tag{44}$$

where N is the number of Monte-Carlo simulations; j is the jth simulation; x and y respectively represent the true value and filtered estimated value of the tracking target state at time k.

In addition, the velocity RMSE is treated by taking relative method, and the formula is as follows:

$$\mathrm{RMSE}(v)=\frac{\sqrt{\frac{1}{N}\sum_{j=1}^{N}\left(\boldsymbol{X}(k)-\hat{\boldsymbol{X}}^{j}(k|k)\right)^{2}}}{\sqrt{v_1^2+v_2^2}}. \tag{45}$$

In (45), the square root values of the two dimensions of velocity are calculated at each moment, and then the value is used to normalize the RMSE of velocity.

Scenario 1 Target turning maneuver: The target movement process lasts 400 s, initial state: (1 000,10, 0,1 000,10,0). The target starts to move in a straight line at a constant velocity (CV), with an initial speed of 10 m/s. It then executes a maneuver with constrant acceleration (CA) of 1 m/s^2 from 101 s−191 s. Then it executes a constant turning (CT) motion in 191 s−270 s, and the centripetal acceleration is $-\pi/270\ \mathrm{m\cdot s^{-2}}$. Finally, it goes back to constant velocity motion. The target trajectory is shown in Fig. 3.

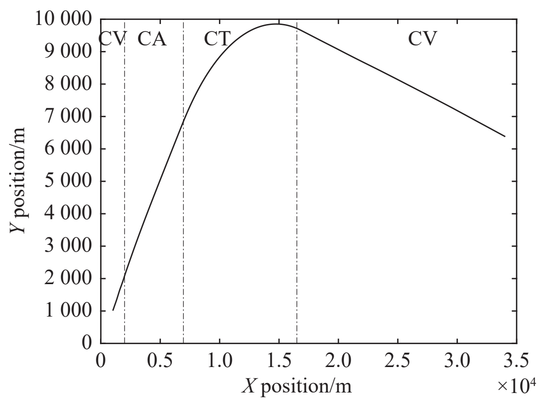

Fig. 3 Track diagram in Scenario 1

It can be seen from Fig. 4 and Fig. 5 that in the case of turning maneuver, the RMSE of position and speed of the improved IMM algorithm is obviously lower than that of the IMM algorithm. Since the target moves at a constant speed in the initial stage, the model set can match it, the position and velocity RMSEs of the two algorithms are relatively low. In 101 s−190 s, the target moves at a constant acceleration, and the model sets cannot match. Therefore, the RMSE of the position and velocity of the two algorithms increases greatly, and reaches the peak after a period of time, but the improved IMM algorithm is lower than the IMM algorithm. In 191 s−270 s, the target makes a constant turning motion and the model sets match, and the RMSE of the position and velocity errors of the two algorithms begins to decrease. In 271 s−400 s, the target moves at a constant speed and the model sets match, and the RMSE of position and velocity continues to decrease and then floats at a lower level. Under the premise of retaining the low RMSE when the model is matched, it can be seen that the improved IMM algorithm can effectively reduce the RMSE when the model does not match. Finally, the variation trend of adaptive parameters is shown in Fig. 6.

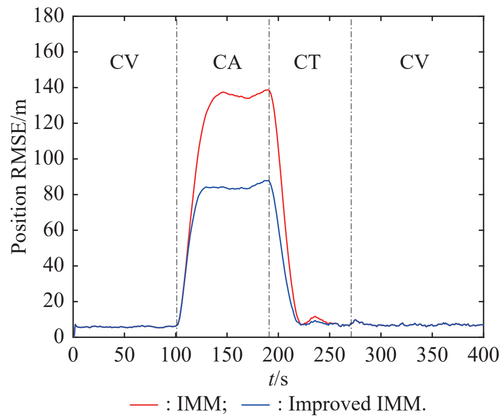

Fig. 4 RMSE of position in Scenario 1

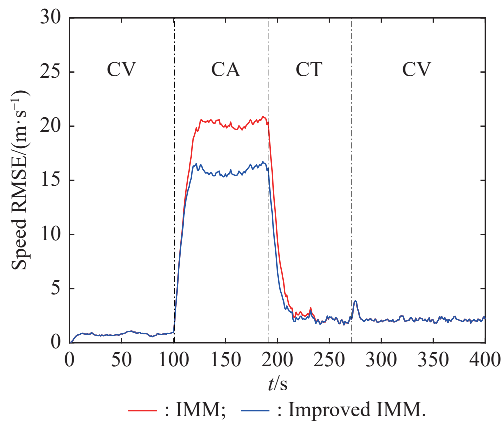

Fig. 5 Root mean square error of speed in Scenario 1

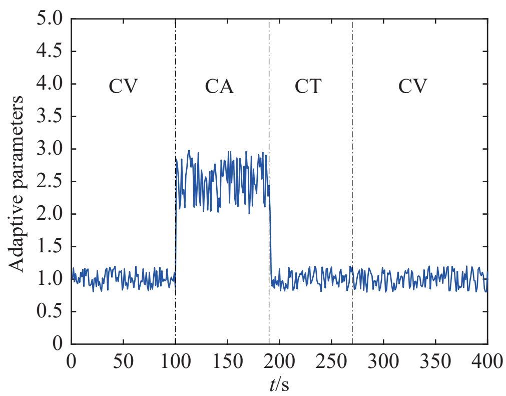

Fig. 6 Adaptive parameter changes in Scenario 1

Scenario 2 Target continuous maneuver: The target starts to move in a straight line at a constant velocity, with an initial speed of 10 m/s. It then executes a maneuver with an acceleration of 1 m/s^2 from 51 s−100 s. Then it executes a constant turning motion in 101 s−300 s, and the centripetal acceleration is $-\pi/270\ \mathrm{m\cdot s^{-2}}$. After that, conduct constant acceleration motion in 300 s−350 s,

X-axis acceleration is 1 $\mathrm{m \cdot s^{-2}}$ and Y-axis acceleration is -1 $\mathrm{m \cdot s^{-2}}$. Finally, it goes back to constant velocity motion. The target trajectory is shown in Fig. 7.

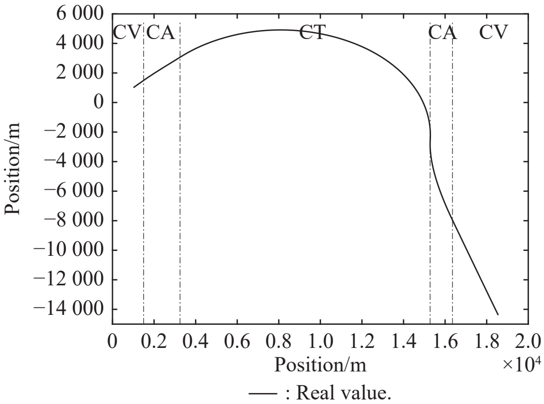

Fig. 7 Track diagram in Scenario 2

It can be seen from Fig. 8 and Fig. 9 that when the target is continuously manipulated, the improved IMM algorithm can effectively reduce the RMSE when the model does not match. At the same time, the RMSE is kept small when the model is matched. Since the target moves at a constant speed in the initial stage and the model sets match, the position and the velocity RMSE of the two algorithms are relatively low. Within 51 s–100 s, the target moves at a constant acceleration, and the model set cannot be matched. Therefore, the RMSE of the position and the velocity of the two algorithms are greatly increased, and they reach a peak after a period of time, but the RMSE of the improved IMM algorithm is lower than that of the IMM algorithm. In 101 s–300 s, the target makes a constant turning motion, and the model set can be matched. Therefore, the RMSE of the position and the velocity errors of the two algorithms begins to decrease. Within 300 s–350 s, the target moves at a constant acceleration and the model set does not match, so the RMSE of the position and the velocity rises again. Within 350 s–400 s, the target moves at a constant speed and the model set matches, so the RMSE of the position and the speed drops again. Finally, the variation trend of adaptive parameters is shown in Fig. 10.

The above two examples illustrate that the improved IMM algorithm mainly plays the following roles.

(i) When the model is matched, the tracking accuracy of the traditional IMM algorithm is guaranteed.

(ii) When the model does not match, the feedback network adjusts the degree of trust to the measured value, corrects the covariance prediction value in real time, and eliminates or reduces the filtering tracking accuracy drop and filtering divergence caused by the sudden change of the target state.

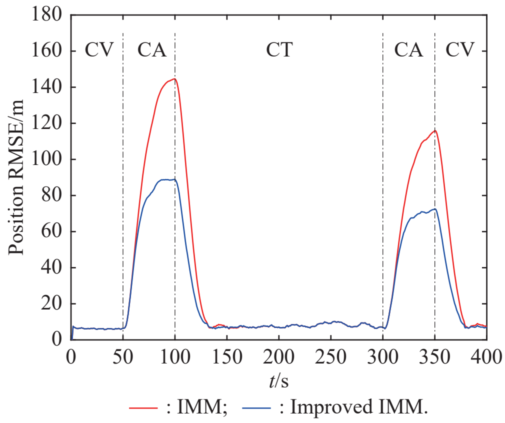

Fig. 8 RMSE of position in Scenario 2

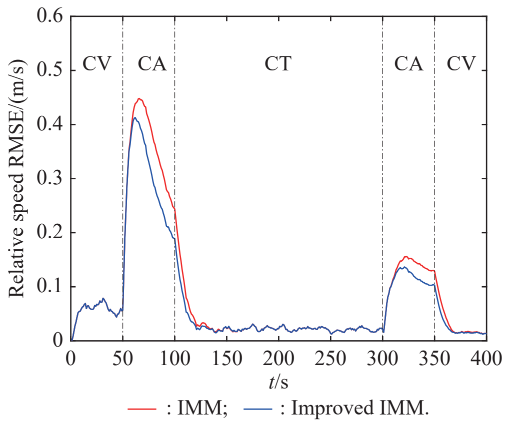

Fig. 9 RMSE of speed in Scenario 2

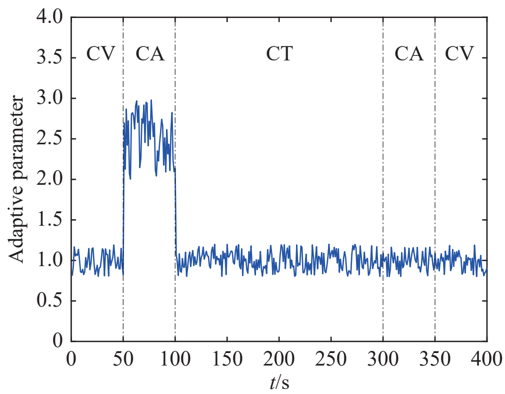

Fig. 10 Adaptive parameter changes in Scenario 2

4. Conclusions

This paper proposes an improved IMM algorithm based on support vector regression. The traditional IMM algorithm, the SVR method and the improved IMM method based on the above are discussed.

Two types of scenarios are simulated for the traditional IMM algorithm and the improved IMM algorithm. From the experimental results, the following conclusions can be drawn. By using SVR to improve the traditional

IMM algorithm, the estimated RMSE of the traditional IMM algorithm in the case of model mismatch is reduced, and the adaptability of the algorithm is improved. The ideas presented in this paper can also optimize and improve single model tracking algorithms and other multi-model tracking algorithms.

References

[1] BLOM H, BAR-SHALOM Y. The interacting multiple model algorithm for systems with Markovian switching coefficients. IEEE Trans. on Automatic Control, 1988, 33(8): 780–783.

[2] HE Y, XIU J J, GUAN X. Radar data processing and application. Beijing: Publishing House of Electronics Industry, 2009. (in Chinese)

[3] MAGILL D T. Optimal adaptive estimation of sampled stochastic processes. IEEE Trans. on Automatic Control, 1965, 10: 434–439.

[4] ELLSWORTH D A. A new algorithm for interactive graphics on multicomputers. IEEE Computer Graphics & Applications, 1994, 14(4): 33–40.

[5] LI X R, ZHANG Y M, ZHI X R, et al. Multiple-model estimation with variable structure Part IV: design and evaluation of model-group. IEEE Trans. on Aerospace & Electronic Systems, 1999, 35(1): 242–254.

[6] HAN B, HUANG H Q, LEI L, et al. An improved IMM algorithm based on STSRCKF for maneuvering target tracking. IEEE Access, 2019, 7: 57795–57804.

[7] ZHANG R, HU G P. Target tracking algorithm based on IMM/MSPDAF data fusion of multi-sensor. Modern Defence Technology, 2010, 38(6): 123–127. (in Chinese)

[8] KIM B D, LEE J S. IMM algorithm based on the analytic solution of steady state Kalman filter for radar target tracking. Proc of the IEEE International Radar Conference, 2005. DOI: 10.1109/RADAR.2005.1435927.

[9] XIONG Z K, JANG H Y, LI Q, et al. The research of maneuvering target tracking based on interacting multiple model. Proc. of the IET International Conference on Automatic Control and Artifical Intelligence, 2013. DOI: 10.1049/cp.2012.1429.

[10] CHEN X, LI Z W, HU X D, et al. Research on radar/infrared data fusion target tracking algorithm in cluttering. Journal of Chinese Computer Systems, 2019, 40(8): 1794−1798. (in Chinese)

[11] GOSWAMI A, LEE C S G. Design of an interactive multiple model based two-stage multi-vehicle tracking algorithm for autonomous navigation. Proc. of the IEEE Intelligent Vehicles Symposium, 2015(10): 261–266.

[12] TIAN J L, FU C Y, TANG T. Maneuver-adaptive target tracking algorithm with bearings-only measurements. Opto-Electronic Engineering, 2011, 38(10): 57–65. (in Chinese)

[13] LI W L, JIA Y M. An information theoretic approach to interacting multiple model estimation. IEEE Trans. on Aerospace & Electronic Systems, 2015, 51(3): 1811–1825.

[14] SEAH C E, HWANG I. Algorithm for performance analysis of the IMM algorithm. IEEE Trans. on Aerospace & Electronic Systems, 2011, 47(2): 1114–1124.

[15] OSBORNE R W, BLAIR W D. Update to the hybrid conditional averaging performance prediction of the IMM algorithm. IEEE Trans. on Aerospace & Electronic Systems, 2011, 47(4): 2967–2974.

[16] GONG S L, WANG X H, HUANG S G. Tracking moving target on airport surface based on variable-structure IMM algorithm. Advanced Materials Research, 2012, 459: 603–608.

[17] SHIN H J, EOM D H, KIM S S. One-class support vector machines-an application in machine fault detection and classification. Computers & Chemical Engineering, 2005, 48(2): 395–408.

[18] DIEGO F F, DAVID M R, FONTENLA-ROMERO O, et al. Automatic bearing fault diagnosis based on one-class v-SVM. Computers & Industrial Engineering, 2013, 64(1): 357–365.

[19] XIE S Q, SHEN F M, QIU X N. Face recognition method based on support vector machine. Computer Engineering, 2009, 35(16): 186–188. (in Chinese)

[20] ATHERTON D P, LIN H J. Parallel implementation of IMM tracking algorithm using transputers. IEE Proceedings-Radar, Sonar and Navigation, 2002, 141(6): 325–332.

[21] LI X R, JILKOV V P. A survey of maneuvering target tracking-Part III: measurement models. Proceedings of SPIE - The International Society for Optical Engineering, 2001, 4473: 423–446.

Biographies

ZENG Yuan was born in 1979. She received her B.S. degree in Hunan Normal University, China, in 2001, and M.S. degree in Wuhan Institute of Physics and Mathematics, China, in 2004, and Ph.D. degree in Shanghai Institute of Microsystem and Information Technology (SIMIT) of the Chinese Academy of Sciences in 2009. From 2009 to 2014, she worked as an assistant professor in SIMIT. She is a senior engineer with Shanghai Spaceflight Electronic and Communication Equipment Research Institute. Her main research interests include signal processing and radar technology.

E-mail: iamzengyuan@hotmail.com

LU Wenbin was born in 1980. He received his B.S. degree in electronic engineering from Xidian University in 2003 and M.S. degree in electronic circuit and system from Nanjing University of Aeronautics and Astronautics in 2006. He has been with Shanghai Spaceflight Electronic and Communication Equipment Research Institute. His main research interests include radar technology and signal processing.

E-mail: dawen_lu@126.com

YU Bo was born in 1996. He received his B.S. degree in electronic information engineering form Tianjin University of Technology, Tianjin, China, in 2018. He is currently pursuing his M.S. degree in electronics and communication engineering with Nanjing University of Science and Technology, Nanjing, China. His current research interests include machine learning and radar data processing.

E-mail: yb102725@njust.edu.cn

TAO Shifei was born in 1987. He received his B.S. and Ph.D. degrees from the Department of Communication Engineering, Nanjing University of Science and Technology (NJUST), Nanjing, China, in 2008 and 2014, respectively. Since 2017, he has been with NJUST, and now he is an associate professor in the Department of Communication Engineering, NJUST. From 2015 to 2016, he was a postdoctoral research associate in electronic and computer engineering in Northeastern University, Boston, USA. His current research interests include the electromagnetic theory and antenna technology, and SAR images processing.
E-mail: s.tao@njust.edu.cn

ZHOU Haosu was born in 1989. He received his B.S. degree in communication engineering from Nanjing University of Science and Technology in 2012 where he received his Ph.D. degree in information and communication engineering in 2018. Since April 2018, he has been with Shanghai Spaceflight Electronic and Communication Equipment Research Institute, where he is now an engineer. His main research interests include signal processing, digital communications, indoor position system, automatic identification system, and very high frequency data exchange system.
E-mail: zhouhaosu@sina.com

CHEN Yu was born in 1980. He received his B.S. degree in electronic engineering in 2003 from Xidian University. He received his M.S. degree in 2006 from National University of Defense Technology and has been working in Science and Technology on Near-Surface Detection Laboratory since September 2015. He is now an associate researcher. His research interests include signal processing and automatic recognition.
E-mail: cy0520tool@sohu.com.cn