Chinese Journal of Electronics
Vol.32, No.5, Sept. 2023

An Adaptive Interactive Multiple-Model Algorithm Based on End-to-End Learning

ZHU Hongfeng, XIONG Wei, and CUI Yaqi

(*Institute of Information Fusion, Naval Aviation University, Yantai 264001, China*)

Abstract — The interactive multiple-model (IMM) is a popular choice for target tracking. However, to design transition probability matrices (TPMs) for IMMs is a considerable challenge with less prior knowledge, and the TPM is one of the fundamental factors influencing IMM performance. IMMs with inaccurate TPMs can make it difficult to monitor target maneuvers and bring poor tracking results. To address this challenge, we propose an adaptive IMM algorithm based on end-to-end learning. In our method, the neural network is utilized to estimate TPMs in real-time based on partial parameters of IMM in each time step, resulting in a generalized recurrent neural network. Through end-to-end learning in the tracking task, the dataset cost of the proposed algorithm is smaller and the generalizability is stronger. Simulation and automatic dependent surveillance-broadcast tracking experiment results show that the proposed algorithm has better tracking accuracy and robustness with less prior knowledge.

Key words — Maneuvering target tracking, Neural networks, End-to-end learning, Interactive multiple-model, Model transition probability matrix.

I. Introduction

The maneuvering target tracking has been widely investigated and applied in both military and civilian fields, such as early warning [1] and aircraft traffic control (ATC) [2]. Due to the complexity and uncertainty of maneuvering target motion, model-based target tracking algorithms cannot achieve satisfactory and stable performances [3], [4]. The interacting multiple-model (IMM) [5] algorithm based on the Markov jump linear system is one of the most effective methods in the field of maneuvering target tracking. There have been wide applications of IMM in diverse fields, and many advanced IMM techniques are also being studied [6], [7].

Classic IMM algorithms adopt the homogeneous Markov chain to govern the model transition, and transition probability matrices (TPMs) are constant matrices [8]. However, it is very difficult to obtain sufficient prior knowledge in practical applications. There are three dilemmas around TPMs. First, it is not possible to calculate the corresponding TPM for each trajectory independently. Second, especially in military applications, maneuvers of targets are generally arbitrary, unpredictable and unexpected, so their transition probability matrixes should also be time-varying. Third, in terms of the influence of the TPM on the iterative process of the IMM, the larger the probability of the model transfer to itself is, the larger the matching model probability calculated and the higher the tracking accuracy are in the tracking stable region. But it will make the model inertia increase, which results in the cost of model switching and peak errors increase even tracking diverges. So, there is a contradiction between the stable tracking accuracy of the target and the peak error of maneuvering switching. If the preset TPMs deviate from the true value too much, the performance of IMM will decrease dramatically.

Nowadays, multiple-model algorithms have been developed to the third generation, namely variable structure with multiple-model estimation (VSMM) [9]. VSMM has three main model-set adaptive strategies: the model-group switching [10], likely mode set [11], and expected-mode augmentation [12], which are all based on the IMM framework. That is, the model interaction and estimate fusion in each VSMM model set are governed by the TPMs. And the adaption or switching of model sets in VSMM is directly influenced by TPMs. Furthermore, adaptive algorithms of IMM and VSMM are also based on the standard IMM framework, such as

Manuscript Received Dec. 22, 2021; Accepted July 3, 2022. This work was supported by the Major Program of the National Natural Science Foundation of China (61790554) and the Youth Program of National Natural Science Foundation of China (62001499).
 DOI:10.23919/cje.2021.00.442

the reweighted IMM (RIMM) [13], switched IMM-extended Viterbi [14], and hybrid grid multiple-model (HGMM) [15]. In other words, the basic performance of the standard IMM algorithm affects all adaptive IMM algorithms, VSMM, and their enhancement algorithms. If TPMs are more accurate, all the above algorithms will benefit from it and further improve their performance.

For the aforementioned drawbacks of the TPM of the IMM, many researchers have done fruitful explorations in this field. When the target maneuver occurs, model likelihoods and model probabilities of the IMM start to change, then models start to switch. These studies can be summarized as how to model the mapping function from model probabilities and/or likelihoods to TMP. References [16], [17] adaptively estimate TPMs by constructing the model error compression rate (obtained from the TPM elements and model probabilities), which increases the accuracy of the model stabilization stage. But the robustness of the algorithm decreases due to over-regulation of TPM. References [18], [19] use an exponential function of the model probability change rate to map the adjustment rate of the TPM elements, and the algorithm is more stable, but the adjustment is not deep enough. Reference [20] calculates the TPM based on the likelihood ratio of models, but only two models are considered in the paper. Reference [21] models the TPM correction function by using model probability gradients and ratio of likelihood, and two parallel IMMs are used to get more information about model jumping. But the parallel structure doubles the computational effort and does not validate more than two models. However, there are same defects of above methods of TPM adaptation, the functions of TMP estimation are constructed empirically and additional empirical parameters (e.g., threshold) are introduced, so that the reliability of the algorithm cannot be ensured. Moreover, adjustment mechanisms are not deeply coupled with IMM, which decrease the robustness of the IMM, and result in divergence because of misadjustment.

Deep learning and neural networks have been developed at a high speed in the last decade, with extensive applications and impressive results in many fields such as computer vision, language processing, and automation control [22], [23]. Although neural networks can model complex nonlinear functions, there are practical difficulties to apply deep learning in the TPM adaptation task directly, such as no explicit training dataset and large scale of data noise.

In this paper, an adaptive IMM algorithm based on end-to-end learning (EEL-IMM) is proposed, where the TPM is estimated by one neural network, resulting in a generalized recurrent neural network is obtained. The EEL-IMM takes improving tracking accuracy as the learning goal and learns the estimation function of TPM end-to-end in a real-time tracking task. By this means the EEL-IMM can avoid using the TPM dataset directly and reduce the cost of the dataset. And the coupling between neural network and IMM is enhanced, which improves the robustness and environmental resilience of the IMM. The rest of this paper is organized as follows. Section II is the problem formulation. In Section III, the EEL-IMM algorithm and its training process are described in detail. Then, simulation experiments and analysis in two prior scenarios and the automatic dependent surveillance-broadcast (ADS-B) dataset tracking are shown in Section IV. Section V concludes the paper.

II. Problem Formulation

1. TPM adaptive adjustment analysis

The hybrid state space model used in this paper for the maneuvering target tracking problem is

$$\boldsymbol{x}(k+1)=\boldsymbol{F}(k)\boldsymbol{x}(k)+\boldsymbol{G}(k)\boldsymbol{\varepsilon}(k) \tag{1}$$

$$\boldsymbol{z}(k+1)=\boldsymbol{H}(k+1)\boldsymbol{x}(k+1)+\boldsymbol{w}(k+1) \tag{2}$$

where $\boldsymbol{x}(k+1)$ and $\boldsymbol{z}(k+1)$ are the state estimation and measurement at time $k+1$. $\boldsymbol{F}(k)$, $\boldsymbol{H}(k+1)$, and $\boldsymbol{G}(k)$ are the state transition matrix, measurement matrix, and noise transition matrix, respectively. Process noise vector and measurement noise vector are denoted by $\boldsymbol{\varepsilon}(k)$ and $\boldsymbol{w}(k+1)$, respectively, which are uncorrelated zero-mean white Gaussian processes.

Assuming that the number of models used in IMM is N, the models set is $\mathbb{M}=\{M_1,M_2,\ldots,M_i,\ldots,M_N\}$ and $i=1,2,\ldots,N$, then the model transition probability matrix (TPM) $\boldsymbol{\Pi}$ is defined as

$$\boldsymbol{\Pi}=\begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1N} \\ p_{21} & p_{22} & \cdots & p_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ p_{N1} & p_{N2} & \cdots & p_{NN} \end{bmatrix}, \quad \sum_{j=1}^{N} p_{ij}=1 \tag{3}$$

The steps of the IMM algorithm are summarized as follows [6]:

1) Model interaction:

$$\hat{\boldsymbol{X}}^{oj}(k-1)=\sum_{i=1}^{N}\hat{\boldsymbol{X}}^{i}(k-1)u_{k-1|k-1}(i|j) \tag{4}$$

$$\begin{cases} u_{k-1|k-1}(i|j)=\boldsymbol{\Pi}_{ij}(k-1)\,u_{k-1}(i)/\bar{C}_j \\ \bar{C}_j=\sum\limits_{i=1}^{N}\boldsymbol{\Pi}_{ij}(k-1)\,u_{k-1}(i) \end{cases} \tag{5}$$

$$\begin{aligned}&\boldsymbol{P}^{oj}(k-1)\\&=\sum_{i=1}^{N}u_{k-1|k-1}(i|j)\Big\{\boldsymbol{P}^{i}(k-1)\\&\quad+[\hat{\boldsymbol{X}}^{i}(k-1)-\hat{\boldsymbol{X}}^{oj}(k-1)]\,[\hat{\boldsymbol{X}}^{i}(k-1)-\hat{\boldsymbol{X}}^{oj}(k-1)]^{\mathrm{T}}\Big\}\end{aligned}\tag{6}$$

where $\hat{\boldsymbol{X}}^{i}(k-1)$ is the state estimation of model i at time $k-1$, and its error covariance matrix is $\boldsymbol{P}^{i}(k-1)$, u_{k-1} and $u_{k-1|k-1}$ are the model probability vector and the mixed probability matrix, respectively. The mixed state estimation of model j at time $k-1$ is denoted by $\hat{\boldsymbol{X}}^{oj}(k-1)$, and its error covariance matrix is denoted by $\boldsymbol{P}^{oj}(k-1)$.

2) Model filtering: $\hat{\boldsymbol{X}}^{oj}(k-1)$, $\boldsymbol{P}^{oj}(k-1)$, and the measurement $\boldsymbol{Z}(k)$ at time k are as inputs to filter $j=1,2,\ldots,N$ parallel models, and then $\hat{\boldsymbol{X}}^{j}(k)$, $\boldsymbol{P}^{j}(k)$, the residual vector $\boldsymbol{\nu}_k^j$, and its covariance $\boldsymbol{S}_k^j$ are calculated.

3) Model probability update: the probability $u_k(j)$ of model j at time k is updated by the likelihood Λ_k^j of model j as follows:

$$\Lambda_k^j=\frac{1}{\sqrt{\left|2\pi\boldsymbol{S}_k^j\right|}}\exp\left[-\frac{1}{2}\left(\boldsymbol{\nu}_k^j\right)^{\mathrm{T}}\left(\boldsymbol{S}_k^j\right)^{-1}\boldsymbol{\nu}_k^j\right]\tag{7}$$

$$u_k(j)=\frac{\Lambda_k^j\bar{C}_j}{\sum_{i=1}^{N}\Lambda_k^i\bar{C}_i}\tag{8}$$

4) Model output: at time k, the state estimation $\hat{\boldsymbol{X}}(k)$ and its error covariance matrix $\boldsymbol{P}(k)$ of IMM are

$$\hat{\boldsymbol{X}}(k)=\sum_{i=1}^{N}\hat{\boldsymbol{X}}^{i}(k)u_k(i)\tag{9}$$

$$\begin{aligned}\boldsymbol{P}(k)=&\sum_{i=1}^{N}u_k(i)\Big\{\boldsymbol{P}^{i}(k)\\&+[\hat{\boldsymbol{X}}^{i}(k)-\hat{\boldsymbol{X}}(k)]\,[\hat{\boldsymbol{X}}^{i}(k)-\hat{\boldsymbol{X}}(k)]^{\mathrm{T}}\Big\}\end{aligned}\tag{10}$$

According to the above one circle of IMM, the following qualitative analysis is done. Suppose the matching model of the target at time $k-1$ is M_1. If the target does not maneuver at time k, the probability $p_{j1}(j=1,2,\ldots,N)$ of other models transferring to M_1 should be increased, and the probability p_{1j} of M_1 transferring to other models should be decreased. In this way, the probability u_k^1 of M_1 can be increased or remained at a high level, which makes the tracking accuracy increase. If the target maneuvers at the time k, and the matching model at time k is M_2, in order to switch the model and make the u_k^2 increase as soon as possible, then p_{j2} should be increased, and p_{2j} should be decreased at the same time.

IMMs essentially switch models according to model likelihoods $\boldsymbol{\Lambda}_k$, and estimates u_k. Besides, $\boldsymbol{\Lambda}_k$ and u_k have to be calculated based on the model probability u_{k-1} and the TPM $\boldsymbol{\Pi}_{k-1}$ at the last moment. In other words, the changes of $\boldsymbol{\Lambda}_k$ and u_{k-1} can indicate the maneuver of the target. And TPM needs to be adjusted according to the target maneuver to make IMM adaptable to the scenario. Hence, $\boldsymbol{\Pi}(k)$ can be estimated by $\boldsymbol{\Lambda}_k$ and u_{k-1}. Then we define the following function:

$$\boldsymbol{\Pi}(k)=\boldsymbol{f}_{\boldsymbol{\Pi}}(\boldsymbol{\Pi}(k-1),\boldsymbol{\Lambda}_k,u_k)\tag{11}$$

where $\boldsymbol{\Pi}(k-1)$ is used to assist in the estimation of $\boldsymbol{\Pi}(k)$, and to improve the stability of $\boldsymbol{\Pi}(k)$ estimate. Therefore, the key to TPM adaptivity is to model $\boldsymbol{f}_{\boldsymbol{\Pi}}(\cdot)$.

2. Advantages of end-to-end learning

However, in real environments, noise and unexpected target maneuvers make $\boldsymbol{f}_{\boldsymbol{\Pi}}(\cdot)$ difficult to be mathematically modeled as well as to obtain specific function expressions. So TPM has been to be approximated empirically by using other functions in the existing studies. The $\boldsymbol{f}_{\boldsymbol{\Pi}}(\cdot)$ modeled in this way is narrowly applicable to the environment, parameter sensitive, and less stable. In different scenarios, empirically parameterized $\boldsymbol{f}_{\boldsymbol{\Pi}}(\cdot)$ may result in poorer performance even divergence.

Neural networks have a powerful ability to fit nonlinear functions, and theoretically single hidden layer neural networks can approximate continuous functions of arbitrary complexity [24]. For modeling of $\boldsymbol{f}_{\boldsymbol{\Pi}}(\cdot)$, to employ the neural network is a good solution. However, there are four implementation difficulties in practical applications as follows:

1) Dataset acquisition. As already mentioned, it is difficult to estimate the TPM directly. Therefore, it is almost impossible to obtain the labels or truth values of TPM, which can be used to train the neural network directly.

2) Dataset preprocessing. In practice, it is easier for us to obtain the ground-truth of the target trajectories by other means, such as GPS. But parameters such as motion models, acceleration, and velocity of the target are generally unavailable or costly to obtain, which can directly reflect the target maneuvers and intention, and are the main basis for estimating the TPM. And the dataset normalization is also difficult due to the characteristics of the trajectories. So, the cost of dataset preprocessing is large.

3) Neural networks are difficult to deeply couple with tracking algorithms. The external direct training

process of TPM cannot get real-time feedback from the IMM algorithm. With this training approach, it is difficult for the learned $\boldsymbol{f}_{\boldsymbol{\Pi}}(\cdot)$ neural network to understand the numerical fluctuations of IMM-related parameters and output the correct adaptation in actual maneuvering target tracking tasks. Then $\boldsymbol{f}_{\boldsymbol{\Pi}}(\cdot)$ incorrect feedback may lead to rapid deterioration of IMM performance. So, TPM and IMM should be deeply coupled.

4) In the actual target tracking task, we usually pay more attention to the tracking accuracy, and the estimation accuracy of intermediate parameters is relatively unimportant.

Therefore, it is necessary to embed neural networks in IMM to achieve a combination of target tracking tasks and network learning tasks. In this way, the four problems of dataset acquisition, preprocessing, and coupling between neural networks with IMM can be solved simultaneously.

The filters used for the IMM algorithm in this paper is Kalman filter [25]. The iterative process of IMM and Kalman algorithm are both processes of matrix operations. Therefore, the tracking error gradient of IMM can be conducted from the output of IMM to the output of the neural network embedded in the IMM through matrix derivation, and then the neural network can be trained. And now the mainstream deep learning frameworks can implement automatic differentiation of arbitrary scalar valued functions, such as Pytorch [26]. In summary, the end-to-end learning task proposed in this paper is feasible and simple to implement.

III. EEL-IMM Network Structure and Training

1. EEL-IMM network

The EEL-IMM network structure we proposed is shown in Fig.1 according to the analysis in Section II.2.

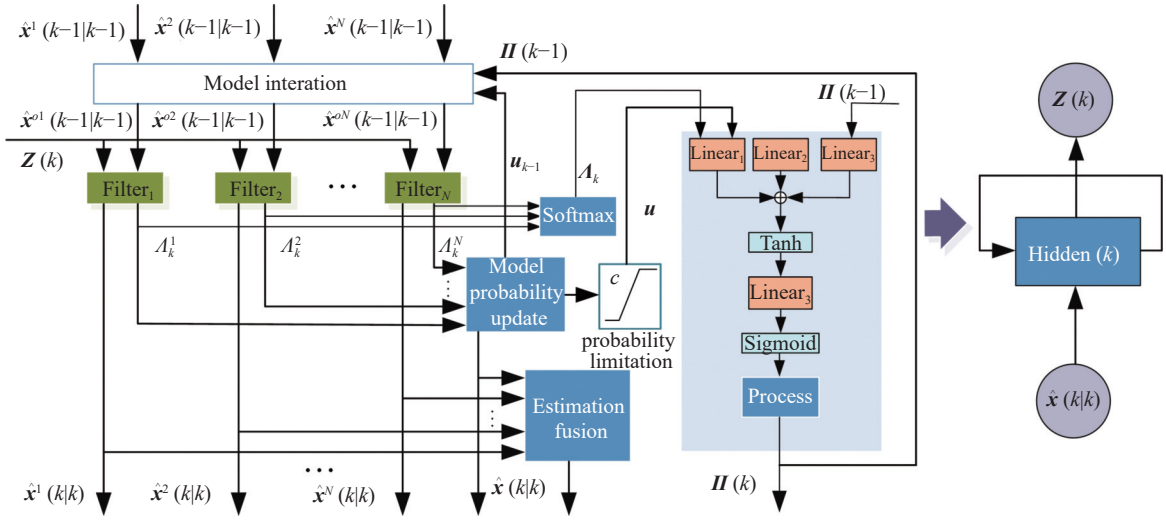


Fig. 1. EEL-IMM algorithm network structure

The calculation of (11) is done using a neural network. The $\text{linear}(\cdot)$ is used to denote the single linear neural network layer, the calculation procedure of $\boldsymbol{\Pi}(k)$ in Fig.1 is as follows:

$$\mathbf{out}_1(k) = \text{linear}_1(\text{flatten}(u_k)) \tag{12}$$

$$\mathbf{out}_2(k) = \text{linear}_2(\text{flatten}(\text{Softmax}(\boldsymbol{\Lambda}_k))) \tag{13}$$

$$\mathbf{out}_3(k) = \text{linear}_3(\text{flatten}(\boldsymbol{\Pi}(k-1))) \tag{14}$$

$$\mathbf{out}_4(k) = \text{linear}_3(\text{Tanh}(\mathbf{out}_1(k)+\mathbf{out}_2(k)+\mathbf{out}_3(k))) \tag{15}$$

$$\boldsymbol{\Pi}^{raw}(k) = \text{reshape}(\text{Sigmoid}(\mathbf{out}_4)) \tag{16}$$

where $\text{flatten}(\cdot)$ is the function that expands matrices into vectors. The $\text{reshape}(\cdot)$ is a function that converts vectors into matrices, and in (16) it converts the neural network output vector into an $N \times (N-1)$ matrix to obtain $\boldsymbol{\Pi}^{raw}(k)$. Tanh and Sigmoid are activation functions. The purpose of such processing is for subsequent normalization and processing to facilitate the embedding of the IMM algorithm.

Assuming that the diagonal dominance ratio of $\boldsymbol{\Pi}$ (i.e., the lowest bound of the main diagonal element of $\boldsymbol{\Pi}$) is ρ, then TPM is estimated in next time as

$$\boldsymbol{\Pi}_{ij}(k) = \begin{cases} \boldsymbol{\Pi}_{ij}^{\text{raw}}(k)\dfrac{1-\rho}{N-1}, & i \neq j \\ 1 - \displaystyle\sum_{l=1}^{N-1} \boldsymbol{\Pi}_{il}^{\text{raw}}(k)\dfrac{1-\rho}{N-1}, & i = j \end{cases} \tag{17}$$

But, the probabilitiy of matching model can be

very close to 1 after a period of time due to the positive feedback adjustment of TPM. One of the characteristics of IMM framework is that the maneuver feedback received by IMM is the model likelihoods weighted by model probabilities and TPM. As a consequence, the probabilitiy of non-matching model can be very close to 0. In this case, the cost and peak error of the EEL-IMM switching model increases. To balance this contradiction, a clipping is introduced to limit u_k. Here, we define a clipping operator $C(\cdot)$, s.t. $u_k^i \le c$, $c \in [0.5, 1]$, $i = 1, 2, \ldots, N$, and c is the upper bound of u_k^i. By clipping u_k, the peak error is reduced and the robustness of the algorithm is improved at the expense of a small amount of accuracy in stable tracking stage.

As can be seen from Fig.1, if $\boldsymbol{z}(k)$ is taken as input, $\hat{\boldsymbol{x}}(k|k)$ as output, and $\boldsymbol{u}_k$, $\boldsymbol{\Pi}(k)$, $\hat{\boldsymbol{X}}^o(k|k)$, $\boldsymbol{P}^o(k|k)$ as hidden information hidden(k), the EEL-IMM algorithm network structure can be considered as a generalized recurrent neural network (RNN) [27]. That is, EEL-IMM is an RNN of composite network structure with one iteration of IMM added at each time step. Recurrent neural networks have powerful sequence processing capability by a simple network structure. Therefore, although the neural network embedded in EEL-IMM is very simple, it increases the complexity of the network in the time dimension just like the recurrent neural network. It enables EEL-IMM to tackle complex tasks with little additional computational cost. For convenience of description, assuming that $\mathrm{IMM}_k(\cdot)$ denotes the iteration of IMM from (4) to (10) at time k, and $\boldsymbol{f}_{\boldsymbol{\Pi}(k)}^{RNN}(\cdot)$ denotes the estimation of TPM from (12) to (17) at time k. Then, one iteration of EEL-IMM algorithm can be briefly described as follows.

1) Iteration of IMM:

$$\mathbf{out}_{\mathrm{IMM}}(k), \hat{\boldsymbol{X}}(k), \boldsymbol{P}(k) = \mathrm{IMM}_k\left(\mathbf{out}_{\mathrm{IMM}}(k-1), \boldsymbol{Z}_k\right) \tag{18}$$

$$\mathbf{out}_{\mathrm{IMM}}(k) = \left\{C(u_k), \boldsymbol{\Lambda}_k, \left\{\hat{\boldsymbol{X}}^i(k)\right\}_{i=1}^N, \left\{\boldsymbol{P}^i(k)\right\}_{i=1}^N, \boldsymbol{\Pi}(k)\right\} \tag{19}$$

2) Estimation of TPM:

$$\boldsymbol{\Pi}(k) = \boldsymbol{f}_{\boldsymbol{\Pi}(k)}^{RNN}\left(C(u_k), \boldsymbol{\Lambda}_k, \boldsymbol{\Pi}(k-1)\right) \tag{20}$$

3) EEL-IMM training: if it's in the training stage, the loss of state estimation is calculated first, and then the EEl-IMM is trained once after accumulating l step losses.

2. Dataset generation and network training

In this paper, we use a simulation dataset to train EEL-IMM (firstly, target trajectories are easy to generate by simulation, and secondly, it is more convenient and intuitive to analyze this algorithm subsequently). The engineering application background of the dataset adopts the civilian aircraft tracking background of [28], which tracks in the (x, y) coordinate, and expands the range of some aircraft motion parameters to deal with possible special scenarios. The state vector of the target is $\left[\begin{array}{cccccc} x_k & \dot{x}_k & \ddot{x}_k & y_k & \dot{y}_k & \ddot{y}_k \end{array}\right]^{\mathrm{T}}$. The EEL-IMM algorithm uses three models [29], which are constant velocity (CV), constant acceleration (CA) and coordinate turn (CT).

In order to reduce the cost of dataset and improve the coverage of dataset to the possible movements of targets, the rules of dataset generation adopted in this paper are summarized in Table 1.

Table 1. Dataset generation rules

Parameters	Value
Sampling interval (s)	0.5
Movement time (s)	100
Maximum velocity (m/s)	± 400
Maximum turn rate (degree/s)	± 15
Maximum turn angle (degree)	± 270
x, y axis initial position (m)	rand $(5000, 200000)$
x, y axis initial velocity (m/s)	rand $(-400, 400)$

In Table 1, rand (a, b) denotes the uniform distribution between a and b. The whole trajectory of one target is divided into four stages. The start and end models of the target are CV, and two models in the middle are random order of CA and CT. This way the dataset contains the maneuver switching of the three models under different motion parameters conditions. It should be noted that, the knowledge network learned is the information of target maneuvers. Hence, the parameters independent of the target motion properties of the data set will not affect the training effect, such as the movement time and initial position.

There are 1000 trajectories generated as the training dataset according to the dataset generation rules in Table 1, which are all plotted in Fig.2.

Once the dataset is obtained, the training of the network can start. To be as realistic as possible, only the positions of the targets are used to calculate the losses of the EEL-IMM network (in a practical environment, only the position information of the target is generally available, and the exact values of velocity, acceleration, and other state information are more difficult to obtain, so that the algorithm can directly reduce the cost of dataset acquisition and preprocessing as much as possible in practical applications). As analyzed in the previous section, the EEL-IMM network is a generalized RNN, so a time window needs to be delineated

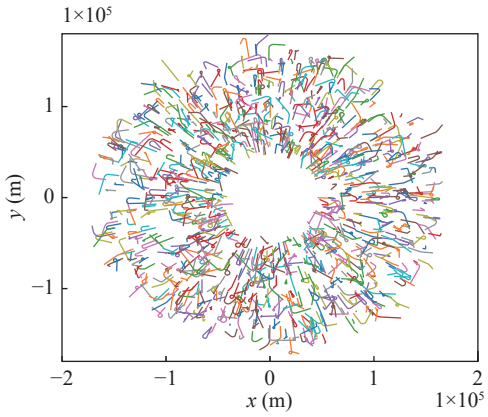


Fig. 2. Trajectories of training dataset

during training to enable the network to learn based on historical information. And the use of time window to train EEL-IMM can also reduce the training time and cost (compared to one-time step training) and improve the stability of training (the time window is no longer needed when the trained network is used). Assuming that the window length is l steps, and the estimating position vector of the target at the jth time window of the ith trajectory is $\hat{\boldsymbol{X}}_p(i,j)$ in one training epoch. $\boldsymbol{X}_p(i,j)$ denotes the true position vector of the target corresponding with $\hat{\boldsymbol{X}}_p(i,j)$. Then the loss of the network at jth time window is defined as:

$$\text{loss}_j = \frac{\sum_{i=1}^{\text{batch}} \sum_{k=1}^{l} \sqrt{\left\| \hat{\boldsymbol{X}}_p(i,j,k) - \boldsymbol{X}_p(i,j,k) \right\|_2}}{\text{batch} \times l} \tag{21}$$

where batch is the batch size. The loss_j is the batch average loss at jth time window in a training epoch, and when a loss_j is calculated, the network is trained and learned once.

For a more detailed analysis of the performance of EEL-IMM, the simulation experiments are carried out in two prior scenarios of an accurate model set and an inaccurate model in this paper. The CT model depends on the turn rate, which is often difficult to know before and during tracking. In this paper, the difference between the two scenarios is whether the prior knowledge of turning rate is obtained. The relevant training parameters of the EEL-IMM model in the two scenarios are set as follows:

Scenario 1 Turning rates are known. The model sets used by EEL-IMM in this scenario consists of CV, CA, and CT (with known turning rates). The standard deviation of process noise is set to 0.1 m for CV and CT, and 1 m for CA due to the large impact of fluctuations in acceleration estimation on tracking. The number of neurons (output dimension size) of the neural network $\text{linear}_{1,2,3}$ is set to 32 (the dimensions of the other two layers depend on the input and output).

Scenario 2 Turning rates are unknown. In this scenario, it is common practice to set the model set covering the target motion patterns as much as possible. So, the EEL-IMM model set consists of CV, CA, CT_{15}, and CT_{-15} (the subscripts indicate that the turning rates are set to 15°, and −15°), and their process noise standard deviations are set to 0.1 m, 1 m, 0.2 m, and 0.2 m, respectively. The number of neurons in the neural network $\text{linear}_{1,2,3}$ is set to 64.

In both scenarios, the TPM diagonal dominance ratios are set to $\rho = 0.5$. The clipping upper bound is set to $c = 0.9$ (The next section carries out a comparison experiment on c). The standard deviation of the measurement noise added to the dataset is 50 m. The batch size used by the network is 1000. The time window size is 10 steps. The number of training epochs are 100, and the initial learning rate is 0.001. The learning rate is decayed by cosine annealing method, and the decay period is 100 epochs. The number of time windows of a trajectory is denoted by nl, the loss of an epoch is calculated as

$$\text{loss}_{\text{epoch}} = \sum_{j=1}^{nl} \text{loss}_j \tag{22}$$

To improve dataset utilization and network generalization, noise is readded to the dataset for each training epoch, enabling the network to better learn and estimate the probability distribution of trajectories through the principles of Monte Carlo method.

Then the EEL-IMM is trained, and the training losses of the two scenarios converge to 27.65 m in scenario 1 and 30.76 m in scenario 2.

IV. Experimental Results and Analysis

1. Experiments on the test dataset

Firstly, the overall tracking performance of EEl-IMM on the test dataset was investigated.

For scenario 1, since the model set is accurate, the main concern is the algorithm's ability to identify maneuvers and to switch models. In scenario 1, algorithms of two different TPM adaptive mechanisms are simulated and compared with EEL-IMM. The first is the algorithm in [16], which adjusts TPM by model error compression rate, and is denoted by AMP-IMM1. The second is the algorithm in [18], which adjusts TPM by the exponent of model probability gradients, and is denoted by AMP-IMM2. AMP-IMM1 and AMP-IMM2 summarize the algorithm principle of [16]–[21] to a certain extent and are both capable of handling scenarios with more than 2 models. And a standard IMM is also used as a comparison algorithm. The model sets, pro-

cess noise, and measurement noise of the above three comparison algorithms are the same as those set in scenario 1 of Section III.2. The TPM of the IMM is

$$\boldsymbol{\Pi}=\begin{bmatrix} 0.9 & p_{N_m} & \cdots & p_{N_m} \\ p_{N_m} & 0.9 & \cdots & p_{N_m} \\ \vdots & \vdots & \ddots & \vdots \\ p_{N_m} & p_{N_m} & \cdots & 0.9 \end{bmatrix},\quad p_{N_m}=\frac{0.1}{N_m-1} \tag{23}$$

and in scenario 1, the size of model set $N_m=3$.

For scenario 2, since the turning rate is unknown, the target motion patterns can only be covered as much as possible by increasing the size of model set. This scenario is to examine the tracking capability of the EEL-IMM algorithm under the inaccuracy of the model set, or the efficiency of model set utilization. Therefore, the main difference between the chosen comparison algorithms is in the set of models, and three comparison algorithms are chosen, namely, IMM-1: $\mathbb{M}_1=\{\mathrm{CV},\mathrm{CA},\mathrm{CT}_{15},\mathrm{CT}_{-15}\}$, IMM-2: $\mathbb{M}_2=\{\mathrm{CV},\mathrm{CA},\mathrm{CT}_{15},\mathrm{CT}_{-15},\mathrm{CT}_{5},\mathrm{CT}_{-5}\}$, and IMM-3: $\mathbb{M}_3=\{\mathrm{CV},\mathrm{CA},\mathrm{CT}_{15},\mathrm{CT}_{-15},\mathrm{CT}_{5},\mathrm{CT}_{-5},\mathrm{CT}_{10},\mathrm{CT}_{-10}\}$, The process noises of the three models are the same as the EEL-IMM algorithm in scenario 2 of Section III.2, and the measurement noise is also the same. TPMs of the three models are set by (23) with $N_{m1}=4$, $N_{m2}=6$, and $N_{m3}=8$, respectively.

There are 200 trajectories generated as a test dataset according to the dataset generation rules of Table 1. The Python 3.7 programming language and Pytorch 1.2 deep learning framework are used. Simulation experiments are conducted to record losses of the four algorithms under (21) and running times of 200 trajectories in Table 2.

Table 2. Comparison of test dataset simulation results

Scenario 1	Results	EEL-IMM	IMM	AMP-IMM1	AMP-IMM2
	Average loss (m)	**27.7557**	32.0487	43.9691	31.4906
	Running time (s)	1.1338	**0.6725**	1.8600	0.6942
Scenario 2	Results	EEL-IMM	IMM-1	IMM-2	IMM-3
	Average loss (m)	**30.8003**	34.9224	34.2543	33.8890
	Running time (s)	1.3515	**0.8946**	1.6858	2.7887

According to Table 2, in both scenarios, in terms of average loss, EEL-IMM has the lowest estimation error on the test dataset and is very close to the lowest training error, indicating that the network is well trained and not overfitted. EEL-IMM has a 4–8 m improvement in tracking accuracy compared to other three compared algorithms. In terms of running time, EEL-IMM has an increased computational cost of 0.4–0.5 s compared to IMM with the same size of model set.

In scenario 1, the model compression rate adjustment of AMP-IMM1 is more complicated, and the calculation process is difficult to matrix (the program matrix runs efficiently), so its running time is the longest. While AMP-IMM2 only calculates the exponents of the model probability gradients, the calculation process is simple, so its running time is only higher than that of IMM. In scenario 2, although EEL-IMM has the coarsest model set, it has the smallest average loss.

2. Experiments of maneuvering target tracking

Secondly, the performance of EEL-IMM in specific maneuvering target tracking tasks was investigated. In this subsection, two different trajectories were randomly selected from the test dataset for the Monte-Carlo experiment of the EEL-IMM. The results of experiments in the previous subsection are interpreted through the tracking results of the EEL-IMM on the single trajectory and the advantages and disadvantages of the EEL-IMM are further analysed.

To evaluate the tracking accuracy, the root mean squared error (RMSE) are chosen as the performance metric, which is defined as

$$\mathrm{RMSE}\,(k)=\sqrt{\frac{\sum_{i=1}^{mc}\left((\hat{\boldsymbol{x}}_i(k)-\boldsymbol{x}_i(k))^2+(\hat{\boldsymbol{y}}_i(k)-\boldsymbol{y}_i(k))^2\right)}{\mathrm{mc}}} \tag{24}$$

where mc is the number of Monte-Carlo simulations, which is 200 in this paper. It should be noted that the RMSE calculated by (24) is larger than the loss calculated by (21). The other values of parameters are the same as those in Section III.1. Then, The main results of experiments are as follows:

1) Results of tracking in scenario 1

Next, position RMSEs of trajectory 1 and the estimated trajectories and their partial enlargements (for easier observing) are depicted in Fig.3 and Fig.4. The curves of model probability estimations are shown in Fig.5.

Similarly, the tracking results of trajectory 2 are illustrated in Figs.6, 7, and 8.

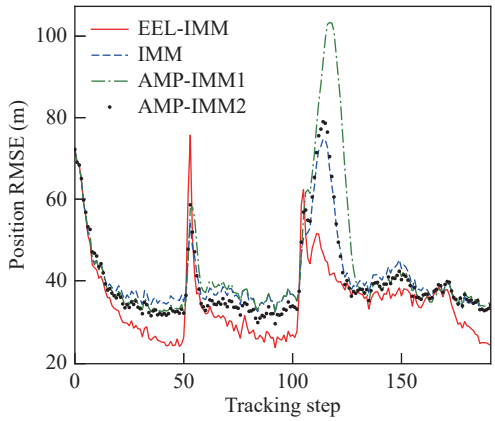


Fig. 3. Position RMSEs of trajectory 1 (scenario 1)

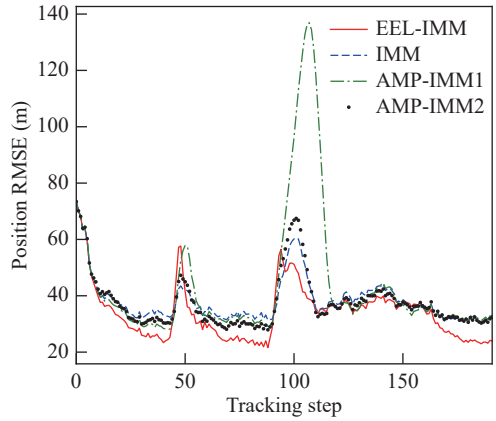


Fig. 6. Position RMSEs of trajectory 2 (scenario 1)

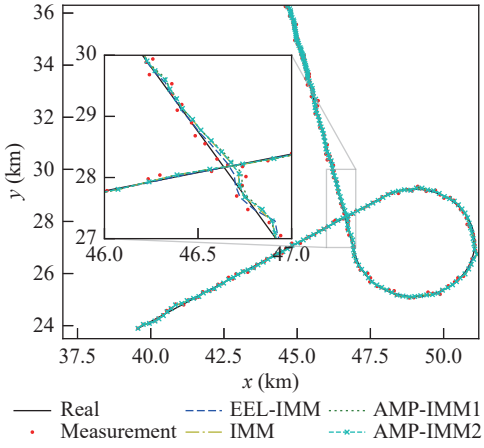


Fig. 4. Estimated trajectories of trajectory 1 (scenario 1). A small area in the grey rectangle is zoomed into this figure for easier observing

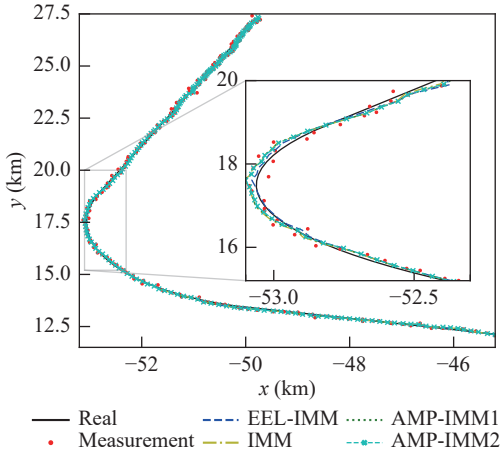


Fig. 7. Estimated trajectories of trajectory 2 (scenario 1). A small area in the grey rectangle is zoomed into this figure for easier observing

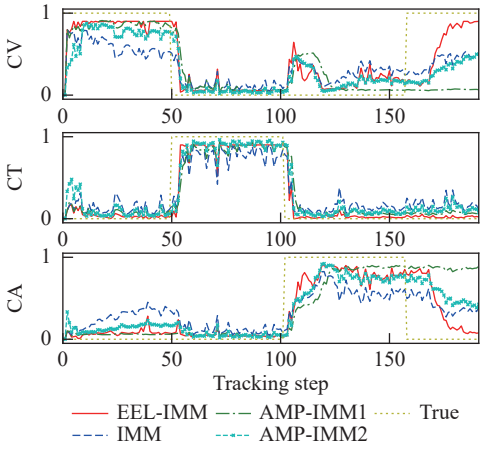


Fig. 5. Estimated model probabilities of trajectory 1 (scenario 1)

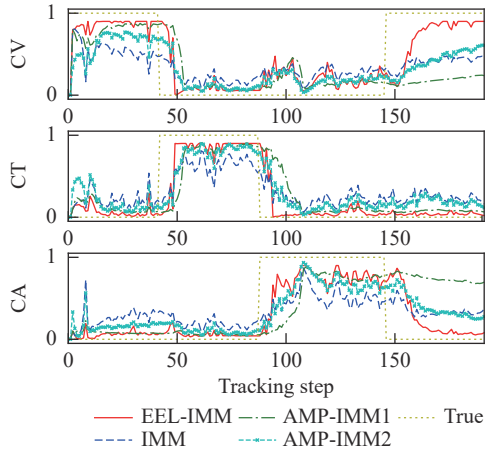


Fig. 8. Estimated model probabilities of trajectory 2 (scenario 1)

The position average RMSE (ARMSE), peak RMSE (PRMSE) and nadir RMSE (NRMSE) are obtained from Monte-Carlo simulation of the two trajectories of scenario 1, and the maximum and minimum values of the estimated model probabilities obtained from one simulation. These metrics are all concluded in Table 3 (to avoid instability at the beginning of the tracking affecting parameters recording, the recording starts from the 10th tracking step). We use Tra., Max-Pr., and Min-Pr. to denote the trajectory, maximum probability, and minimum probability, respectively, so are subsequent tables.

Table 3. Scenario 1 tracking results comparison

Trajectory	Algorithm	ARMSE (m)	PRMSE (m)	NRMSE (m)	Max-Pr.	Min-Pr.
1	EEL-IMM	**33.7656**	**73.2154**	**22.5143**	0.9000	**0.0014**
	IMM	40.1081	75.5026	31.6928	0.8940	0.0341
	AMP-IMM1	42.4123	104.203	30.8557	0.9434	0.0256
	AMP-IMM2	38.8548	79.7740	29.0543	**0.9523**	0.0237
2	EEL-IMM	**32.9518**	**59.2311**	**21.3581**	0.9000	**0.0065**
	IMM	37.4748	60.3196	30.4310	0.8876	0.0370
	AMP-IMM1	43.0800	137.946	25.6648	0.8854	0.0425
	AMP-IMM2	36.7841	67.2878	27.2726	**0.9210**	0.0189

Moreover, the effect of the clipping upper bound c on the performance of the EEL-IMM has also been tested. Here, we carried out experiments for four different values of c on trajectory 1 in scenario 1, and recorded main results in Table 4.

Table 4. Effect of clipping upper bound values on tracking

c	ARMSE (m)	PRMSE (m)	NRMSE (m)	Max-Pr.	Min-Pr.
1.0	**32.8589**	92.8713	**17.2085**	**0.9982**	**0.0005**
0.95	32.6043	81.6468	20.7414	0.9500	0.0012
0.9	33.5707	70.0262	22.1679	0.9000	0.0019
0.85	35.2350	**67.9426**	24.3717	0.8500	0.0020

In scenario 1, the a prior information, i.e. the turning rate, is known and the model set is consistent for all algorithms. As can be seen in Table 3, the ARMSE, PRMSE, NRMSE and minimum probability of EEl-IMM are smallest and respectively reduced by more than 10%, 1.6%, 23%, and 66% as compared with the other three algorithms. Namely, EEL-IMM has the best tracking performance. In Figs.4 and 7, the EEL-IMM has smoother estimated trajectories than that of other algorithms.

Figs.5 and 8 show that the EEL-IMM is able to use the neural network to adaptively tune the TPMs according to the current target state. As a result, the probabilities of matching models are stable at high value and that of non-matching models are stable very low values (they are very close to 0 in Table 3). In other words, the tracking errors of non-matching models have little negative impact on the EEL-IMM. Consequently, as shown in Figs.3 and 6, the tracking accuracy of EEL-IMM increases significantly during stable motion. And the estimated model probabilities of EEL-IMM are very smooth, which shows the EEL-IMM has strong robustness. By contrast, Figs.5 and 8 indicate that, the ways of adjusting TPMs through external empirical function are difficult to bring satisfactory improvement to IMM and even have a negative impact on IMM, such as the performance of AMP-IMM2 is not outstanding because of simple functions, and the AMP-IMM1 switches from CA to CV performed very poorly resulting in highest ARMSE and PRMSE.

However, due to the framework property of IMM, there is a contradiction between model probability and peak error. The higher probabilities of models are, the higher cost of switching the model to other models need, whic can be observed in Table 3, Fig.3, and Fig.6. Tabel 4 numerically details the contradiction between model probabilities and tracking accuracy. The model probability can exceed 0.99, which leads to a large PRMSE. Therefore, we have to sacrifice some tracking accuracy to reduce the peak error. We can find that metrics of tracking in Table 4 decrease (increase) obviously more slowly after 0.9, especially for PRMSE. And when c is 0.9, the PRMSE of EEL-IMM can be reduced to the lowest level compared to the other three algorithms. Based on this fact, c was set at 0.9 in this paper.

2) Results of tracking in scenario 2

The maneuvering target tracking results of the two trajectories in scenario 2 are as follows:

Since the same trajectories are used, only partial enlargements of trajectories are shown here.

Same as Table 3, the ARMSE, PRMSE, NRMSE, Max-Pr. and Min-Pr. of scenario 2 are recorded in Table 5.

In scenario 2, the a prior information, i.e., the turning rate, is not known and the model set of each algorithm is inconsistent. This section experiments with the tracking performance of EEL-IMM in the absence of a prior information. The same advantages and disadvantages of the EEL-IMM as those shown in scenario 1 are not repeated here.

The turning rate of CT model is -11.0671 degree in trajectory 1 and -4.1271 degree in trajectory 2 during 50–100 s. EEL-IMM and IMM1 have the same rough model set, and the preset turning rate of CT model is $\pm 15°$. The CT motion stage of Fig.9 shows that the EEL-IMM can better deal with the situation of unknown model by adaptively combining models as compared with IMM1 and IMM2. That is to say, EEL-IMM has high utilization of model set. But, this advantage of EEL-IMM requires the premise that the model set must not deviate too much from reality, otherwise it will appear as shown in Fig.12.

Table 5. Scenario 2 tracking results comparison

Trajectory	Algorithm	ARMSE (m)	PRMSE (m)	NRMSE (m)	Max-Pr.	Min-Pr.
1	EEL-IMM	**39.1608**	84.3715	**21.8533**	0.9000	**0.0010**
	IMM-1	43.8028	**59.9198**	32.8158	0.9435	0.0079
	IMM-2	43.7844	63.6696	31.7831	**0.9441**	0.0049
	IMM-3	41.9089	63.3020	31.3012	0.7926	0.0037
2	EEL-IMM	**38.1252**	68.8703	**22.5465**	**0.9000**	**0.0021**
	IMM-1	41.3193	**51.4809**	33.2211	0.8384	0.0266
	IMM-2	39.4009	54.6941	32.9292	0.7439	0.0152
	IMM-3	39.6060	54.7306	32.6409	0.7371	0.0097

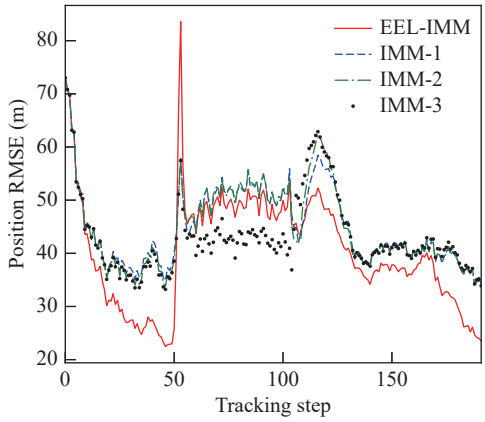


Fig. 9. Position RMSEs of trajectory 1 (scenario 2)

As can be seen from Fig.9–Fig.14, the EEL-IMM can accurately identify existing models in the model set such as CA and CV, despite the interference of unknown models. And probabilities of these identified models remain smoothly at high values, while the probabilities of the other models are close to 0. Additionally, the variable structure multiple-model algorithm or other forms of the model set adaptive algorithms adopt the model probabilities as one of the bases for model set adjustment. Therefore, EEL-IMM can be used instead of IMM as the base framework for VSMM and other model set adaptive algorithms. This can greatly improve the stability and accuracy of VSMM as well as other model

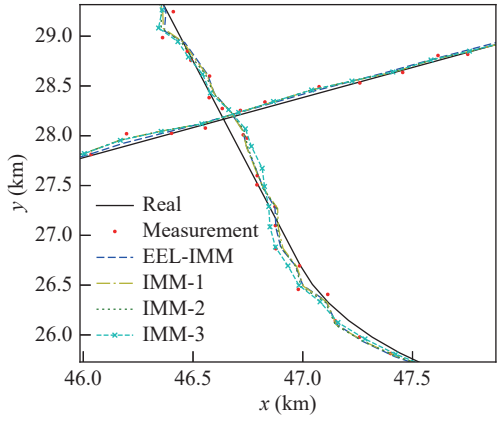


Fig. 10. Partial enlargements of estimated trajectories of trajectory 1 (scenario 2)

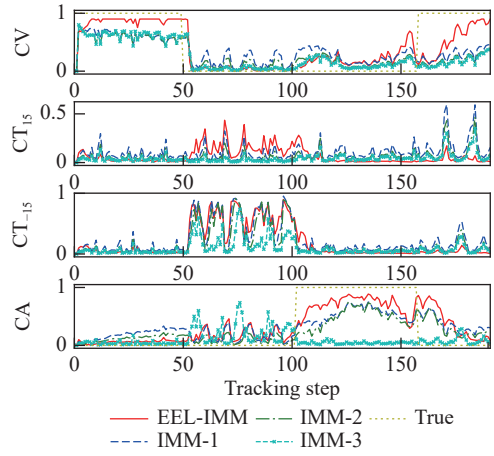


Fig. 11. Estimated model probabilities of trajectory 1 (scenario 2)

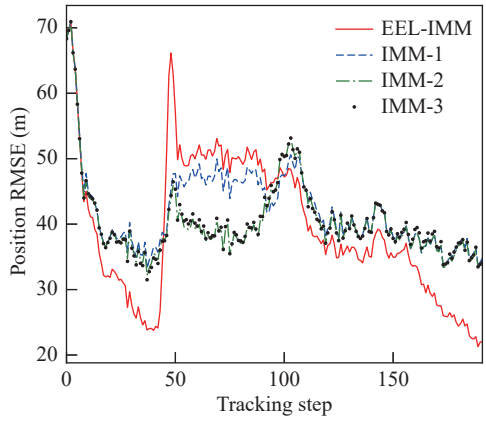


Fig. 12. Position RMSEs of trajectory 2 (scenario 2)

adaptation algorithms, and reduce their design cost due to the advantage of EEL-IMM in estimating the model probabilities.

3. ADS-B trajectories tracking

The previous two subsections conducted experiments on the simulation dataset. This section verifies the tracking effectiveness of EEL-IMM on trajectories of real maneuvering targets The aircraft automatic dependent surveillance-broadcast (ADS-B) dataset is used as the real trajectories. ADS-B data has high accuracy, and besides longitude and latitude, it also has the heading, speed, and other information. In actual maneuvering target tracking, the more difficult tracking is usually in the environment with less measurement informa-

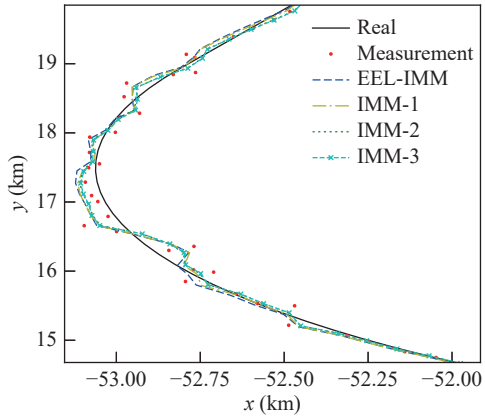


Fig. 13. Partial enlargements of estimated trajectories of trajectory 2 (scenario 2)

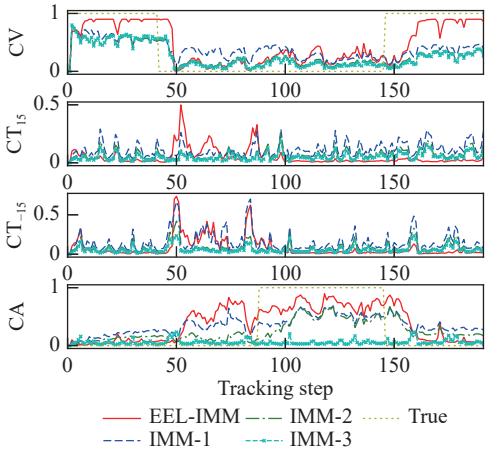


Fig. 14. Estimated model probabilities of trajectory 2 (scenario 2)

tion and larger noise, such as using radar as a tracking sensor. Here, to be closer to the real tracking task, only the aircraft latitude and longitude in ADS-B data are used as true target positions, and then zero-mean white Gaussian noise is added to them as target measurements. For ADS-B data, we do not know some prior information such as motion models of the aircraft. So, the settings of the model set and comparison algorithms are the same as scenario 2 in this subsection.

In this paper, The ADS-B dataset we use is the Weekly 24 hours of State Vector Data public dataset of the OpenSky-Network [30] website. We select the first two packets from the folder of the dataset for May 21, 2021, and preprocess them to obtain 90 trajectories with significant maneuvers. Because the x and y axes have become latitude and longitude, so the process noise of the tracking algorithms needs to change the settings. The standard deviation of the process noise is set to 10^{-2} degree, 10^{-3} degree, and 2×10^{-2} degree for CA, CV, and CT models, respectively. The standard deviation of measurement noise is set to 3×10^{-2} degree. Other settings are the same as in scenario 2.

The trajectories of the pre-processed ADS-B dataset are plotted in Fig.15.

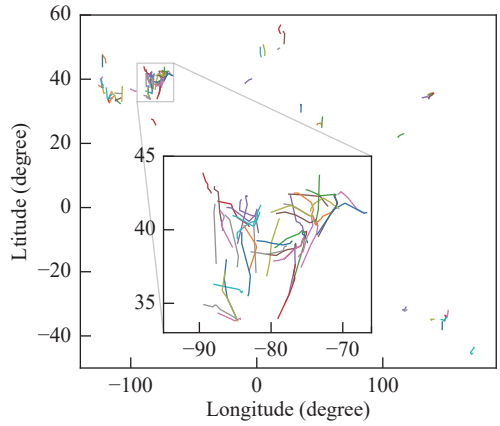


Fig. 15. Trajectories of ADS-B dataset. A small area in the grey rectangle is zoomed into this figure to give more details of the trajectories.

Then, experiments are conducted on the pre-processed ADS-B dataset using EEL-IMM, IMM-1, IMM-2 and IMM-3. Average losses are recorded with Table 6.

Table 6. Comparison of ADS-B dataset tracking results

Results	EEL-IMM	IMM-1	IMM-2	IMM-3
Average loss (degree)	**0.007068**	0.007634	0.007560	0.007535

One trajectory is randomly selected from the ADS-B dataset to conduct the Monte-Carlo experiment, and the experimental results are shown in Figs.16–18 and Table 7.

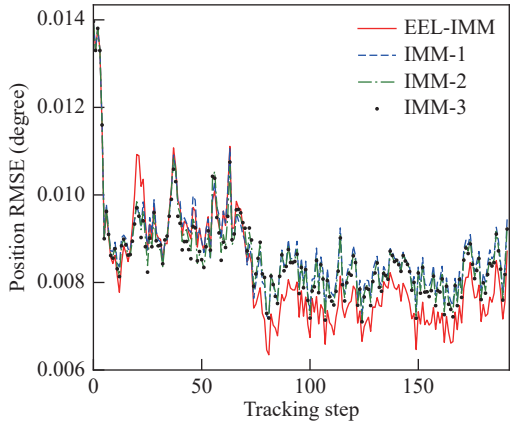


Fig. 16. Position RMSEs of the ADS-B trajectory

It can be seen from Table 6, 7, and Figs.16–18 that EEL-IMM can still maintain advantages analyzed in scenario 1 and scenario 2 on the ADS-B dataset. In other words, EEL-IMM we trained with the simulated dataset can still perform the tracking task of the real trajectories well, which indicates that the EEL-IMM has great environmental adaptability and generaliza-

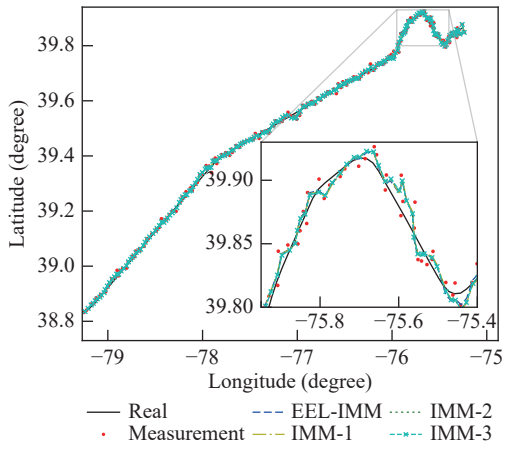


Fig. 17. Estimated trajectories of the ADS-B trajectory. A small area in the grey rectangle is zoomed into this figure for easier observing.

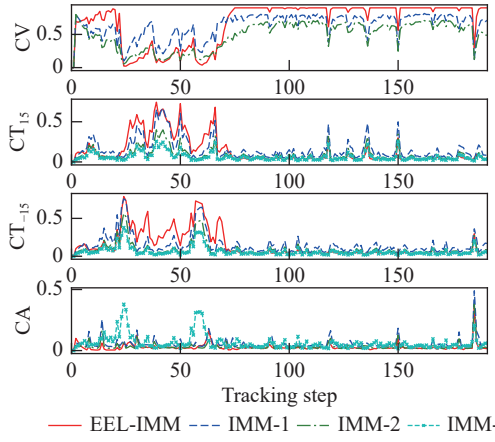


Fig. 18. Estimated model probabilities of the ADS-B trajectory

Table 7. One ADS-B trajectory tracking results comparison

Algorithm	ARMSE (degree)	PRMSE (degree)	PRMSE (degree)	Max-Pr.	Min-Pr.
EEL-IMM	**0.0081**	0.0126	**0.0062**	**0.9000**	**0.0037**
IMM-1	0.0086	0.0115	0.0069	0.8204	0.0254
IMM-2	0.0085	0.0115	0.0068	0.7454	0.0146
IMM-3	0.0084	**0.0113**	0.0067	0.7379	0.0102

tion. In practice, when the real measurement dataset is insufficient or the dataset processing cost is high, we can consider constructing a suitable simulation dataset or a hybrid dataset for EEL-IMM training.

V. Conclusions

In this paper, we propose an adaptive interactive multiple-model algorithm based on end-to-end learning (EEl-IMM) to address the problem for inaccurate transition probability matrices (TPMs) in IMM leading to poor tracking accuracy. The neural network is embedded in the IMM estimate TPMs in real time, resulting in a generalized recurrent neural network (RNN). Thus EEL-IMM we proposed, like RNN, can increase the network complexity in the time dimension and learn complex non-linear functions with only a small increase in computational cost compared to IMM. And EEL-IMM performs end-to-end learning directly in tracking tasks, which improves the robustness of EEL-IMM and avoids the use of inaccessible TPMs datasets. Moreover, We introduce a clipper to clip estimated model probabilities of EEL-IMM, which allows EEL-IMM to sacrifice a small amount of tracking accuracy so that its tracking peak error is controlled within an appropriate range. Simulation results demonstrate that the EEL-IMM proposed in this paper has improved the tracking accuracy, estimated model probabilities accuracy and smoothness, and robustness. The result of experiments on ADS-B trajectories verifies the effectiveness and potentiality of the EEL-IMM for maneuvering target tracking. Future work will include developing an adaptive tracking model to solve the design problem of EEL-IMM model set with insufficiency prior information.

References

[1] H. Wei, Z. P. Cai, B. Tang, *et al.*, "Review of the algorithms for radar single target tracking," *IOP Conference Series: Earth and Environmental Science*, vol.69, article no.012073, 2017.

[2] W. Youn and H. Myung, "Robust interacting multiple model with modeling uncertainties for maneuvering target tracking," *IEEE Access*, vol.7, pp.65427–65443, 2019.

[3] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation: Theory, Algorithms and Software*. John Wiley & Sons, Inc., New York, NY, USA, pp.421–423, 2001, doi: 10.1002/0471221279.

[4] X. R. Li and V. P. Jilkov, "Survey of maneuvering target tracking. Part I. Dynamic models," *IEEE Transactions on Aerospace and Electronic Systems*, vol.39, no.4, pp.1333–1364, 2003.

[5] X. R. Li and Y. Bar-Shalom, "Design of an interacting multiple model algorithm for air traffic control tracking," *IEEE Transactions on Control Systems Technology*, vol.1, no.3, pp.186–194, 1993.

[6] X. R. Li and V. P. Jilkov, "Survey of maneuvering target tracking. Part V. Multiple-model methods," *IEEE Transactions on Aerospace and Electronic Systems*, vol.41, no.4, pp.1255–1321, 2005.

[7] H. Q. Qu, L. P. Pang, and S. H. Li, "A novel interacting multiple model algorithm," *Signal Processing*, vol.89, no.11, pp.2171–2177, 2009.

[8] I. Hwang, C. E. Seah, and S. Lee, "A study on stability of the interacting multiple model algorithm," *IEEE Transactions on Automatic Control*, vol.62, no.2, pp.901–906, 2017.

[9] X. R. Li and Y. Bar-Shalom, "Multiple-model estimation with variable structure," *IEEE Transactions on Automatic control*, vol.41, no.4, pp.478–493, 1996.

[10] X. R. Li, X. Zwi, and Y. Zwang, "Multiple-model estimation with variable structure. III. Model-group switching algorithm," *IEEE Transactions on Aerospace and Electronic Systems*, vol.35, no.1, pp.225–241, 1999.

[11] X. R. Li and Y. Zhang, "Multiple-model estimation with

variable structure. V. Likely-model set algorithm," *IEEE Transactions on Aerospace and Electronic Systems*, vol.36, no.2, pp.448–466, 2000.

[12] X. R. Li, V. P. Jilkov, and J. Ru, "Multiple-model estimation with variable structure - Part VI: Expected-mode augmentation," *IEEE Transactions on Aerospace and Electronic Systems*, vol.41, no.3, pp.853–867, 2005.

[13] L. A. Johnston and V. Krishnamurthy, "An improvement to the interacting multiple model (IMM) algorithm," *IEEE Transactions on Signal Processing*, vol.49, no.12, pp.2909–2923, 2001.

[14] T. J. Ho, "A switched IMM-extended Viterbi estimator-based algorithm for maneuvering target tracking," *Automatica*, vol.47, no.1, pp.92–98, 2011.

[15] L. F. Xu, X. R. Li, and Z. S. Duan, "Hybrid grid multiple-model estimation with application to maneuvering target tracking," *IEEE Transactions on Aerospace and Electronic Systems*, vol.52, no.1, pp.122–136, 2016.

[16] D. C. Dai, M. L. Yao, Z. P. Cai, *et al.*, "Improved adaptive Markov IMM algorithm," *Acta Electronica Sinica*, vol.45, no.5, pp.1198–1205, 2017. (in Chinese)

[17] P. W. Feng, C. Q. Huang, L. P. Cao, *et al.*, "Research on adaptive Markov matrix IMM tracking algorithm," *Systems Engineering and Electronics*, vol.35, no.11, pp.2269–2274, 2010. (in Chinese)

[18] B. Han, H. Q. Huang, L. Lei, *et al.*, "An improved IMM algorithm based on STSRCKF for maneuvering target tracking," *IEEE Access*, vol.7, pp.57795–57804, 2019.

[19] S. Y. Qi, C. D. Qi, and W. H. Wang, "Maneuvering target tracking algorithm based on adaptive Markov transition probabilitiy matrix and IMM-MGEKF," in *Proceedings of 2018 12th International Symposium on Antennas, Propagation and EM Theory (ISAPE)*, Hangzhou, China, pp.1–4, 2018.

[20] J. S. Du and X. Bi, "An adaptive interacting multiple model for vehicle target tracking method," *Advanced Materials Research*, vol.718-720, pp.1286–1289, 2013.

[21] G. Xie, L. L. Sun, T. Wen, *et al.*, "Adaptive transition probability matrix-based parallel IMM algorithm," *IEEE Transactions on Systems, Man, and Cybernetics:Systems*, vol.51, no.5, pp.2980–2989, 2021.

[22] S. Dargan, M. Kumar, M. R. Ayyagari, *et al.*, "A survey of deep learning and its applications: a new paradigm to machine learning," *Archives of Computational Methods in Engineering*, vol.27, no.4, pp.1071–1092, 2020.

[23] M. R. Minar and J. Naher, Recent advances in deep learning: An overview, arXiv preprint arXiv: 1807.08169, 2018, doi: 10.48550/arXiv.1807.08169.

[24] Z. H. Zhou, *Machine Learning*. Tsinghua University Press, Beijing, China, pp.36–40, 2016. (in Chinese)

[25] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Basic Engineering*, vol.82, no.1, pp.35–45, 1960.

[26] A. Paszke, S. Gross, F. Massa, *et al.*, "PyTorch: An imperative style, high-performance deep learning library," in *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, article no.721, Vancouver, Canada, 2019.

[27] W. Zaremba, I. Sutskever, and O. Vinyals, Recurrent neural network regularization, arXiv preprint arXiv: 1409.2329, 2014, doi: 10.48550/arXiv.1409.2329.

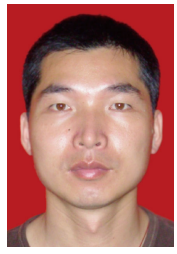
[28] J. X. Liu, Z. L. Wang, and M. Xu, "DeepMTT: A deep learning maneuvering target-tracking algorithm based on bidirectional LSTM network," *Information Fusion*, vol.53, pp.289–304, 2020.

[29] Y. He, J. J. Xiu, and X. Guan, *Radar Data Processing with Applications*, 3rd ed., Publishing House of Electronics Industry, Beijing, China, pp.36–40, 2013. (in Chinese)

[30] M. Schäfer, M. Strohmeier, V. Lenders, *et al.*, "Bringing up OpenSky: A large-scale ADS-B sensor network for research," in *IPSN-14 Proceedings of the 13th International Symposium on Information Processing in Sensor Networks*, Berlin, Germany, pp.83–94, 2014.

ZHU Hongfeng was born in 1994. He received the M.S. degree from Naval Aviation University in 2018, and is currently pursuing the Ph.D. degree in information and communication engineering at Naval Aviation University. His research interests include radar data processing, and target tracking with artificial intelligence.

(Email: 1181630959@qq.com)



XIONG Wei was born in 1977. He received the Ph.D. degree from Naval Aviation University in 2005. He is currently a Full Professor at the Naval Aviation University. He is the Member and Director General of Information Fusion Branch of Chinese Society of Aeronautics and Astronautics. His research interests include radar data processing, multi-sensor information fusion and machine learning.

(Email: xiongwei@csif.org.cn)

CUI Yaqi was born in 1987. He received the Ph.D. degree in information and communication engineering from Naval Aviation University in 2014. He is an Associate Professor at Naval Aviation University. His research interests include information fusion, machine learning, and deep learning with their applications in information fusion.

(Email: cui_yaqi@126.com)