IEEE Journal on Selected Areas in Communications, Available online 30 August 2023
DOI:10.1109/JSAC.2023.3310109

Digital Twins based Intelligent State Prediction Method for Maneuvering-Target Tracking

Jingxian Liu, Junjie Yan, Dehuan Wan*, Xuran Li, Saba Al-Rubaye, *Senior Member, IEEE*, Anwer Al-Dulaimi, *Senior Member, IEEE*, and Zhi Quan, *Senior Member, IEEE*

***Abstract*—Maneuvering-target tracking has always been an important and challenge work because the unknown and changeable motion-models can easily lead to the failure of model-driven target tracking. Recently, many neural network methods are proposed to improve the tracking accuracy by constructing direct mapping relationships from noisy observations to target states. However, limited by the coverage of training data, those data-driven methods suffer other problems, such as weak generalization abilities and unstable tracking effects. In this paper, a digital twin system for maneuvering-target tracking is built, and all kinds of simulated data are created with different motion-models. Based on those data, the features of noisy observations and their relationship to target states are found by two specially designed neural networks: one eliminates the observation noises and the other one predicts the target states according to the noise-limited observations. Combining the above two networks, the state prediction method is proposed to intelligently predict targets by understanding the information of motion-model hidden in noisy observations. Simulation results show that, in comparison with the state-of-the-art model-driven and data-driven methods, the proposed method can correctly and timely predict the motion-models, increase the tracking generalization ability and reduce the tracking root-mean-squared-error by over 50% in most of maneuvering-target tracking scenes.**

* Corresponding author, email: wan_e@gduf.edu.cn

Jingxian Liu is with School of Computer Science and Technology, Guangxi University of Science and Technology, Liuzhou 545006, Guangxi, China, and also with the Liuzhou Key Laboratory of Intelligent Processing and Security of Big Data, Liuzhou, Guangxi, China (e-mail: ljx43031@gxust.edu.cn).

Junjie Yan is with School of Electronic Engineering, Guangxi University of Science and Technology, Liuzhou 545006, Guangxi, China (e-mail: yanjj@gxust.edu.cn).

Dehuan Wan is with Center for Data Science and Artificial Intelligence, Guangdong University of Finance, Guangzhou 510521, China (e-mail: wan_e@gduf.edu.cn).

Xuran Li is with Shandong Key Laboratory of Medical Physics and Image Processing, School of Physics and Electronics, Shandong Normal University, Jinan 250014, China (e-mail: sdnulxr@sdnu.edu.cn).

Saba Al-Rubaye is with the School of Aerospace, Transport and Manufacturing, Cranfield University, Bedford MK43 0AL, U.K (e-mail: S.Alrubaye@cranfield.ac.uk).

Anwer Al-Dulaimi is with Research and Development Department, EXFO, Montreal, QC H4S 0A4, Canada (e-mail: anwer.aldulaimi@ieee.org).

Zhi Quan is with the Guangdong-Hong Kong Joint Laboratory for Big Data Imaging and Communication, Shenzhen, Guangdong 518048, China, and also with the College of Electronics and Information Engineering, Shenzhen University, Shenzhen 518060, China (e-mail: zquan@szu.edu.cn).

This work was supported in part by the National Natural Science Foundation of China under Grant 62061003 and 61971149, in part by the Guangdong Basic and Applied Basic Research Foundation under Grant 2021A1515011657, in part by Liuzhou Science and Technology Plan Project under Grant Numbers 2021ADB0102, in part by the Guangxi Natural Science Foundation under Grant 2022GXNSFBA035642, in part by the Guangdong Provincial Department of Science and Technology, China, under Project No.2020B1212030002, in part by the Shenzhen Science and Technology Innovation Commission, China, and in part by the Basic Ability Enhancement Program for Young and Middle-aged Teachers of Guangxi under Grant 2022KY0325.

Manuscript received October 31, 2022; revised May 18, 2023.

***Index Terms*—Maneuvering-target Tracking, Digital Twins, Neural Network.**

I. Introduction

TARGET tracking refers to predict target states given noisy and information-incomplete observations. Traditional tracking methods, such as Kalman filter (KF) [1], extended KF (EKF) [2], unscented KF (UKF) [3] and particle filter (PF) [4], for tracking are based on the Bayesian tracking framework [5], which predicts the target state depending on a predefined motion-model in each iteration. Obviously, without correct motion-model, the tracking performance cannot be guaranteed.

In most non-cooperative target tracking scenes, targets always move in maneuvering manner. The tricky issue of those maneuvering-target tracking (MTT) is that the motion-models are unpredictable and ever-changing. Hence, it is very difficult to correctly predict the states with proper motion-models. As a result, the tracking performance will severely degrade.

Currently, there are two types of methods to solve the maneuvering-target tracking problems. The first type belongs to the model-driven methods and the second belongs to the data-driven methods. Model-driven methods approximate the motion-model based on combining different known models together. Therefore, the changing patterns of maneuvering-target can be explored. Benefiting from the limitation of known models, those model-driven methods always have good stability for tracking. However, the pattern exploration consumes a lot of time. Those model-driven methods easily fail in high-maneuvering tracking scenes in which the motion-model changes too frequently to be explored timely and correctly.

Different from model-driven method, data-driven methods concentrate on exploiting the features of observations to obtain a direct prediction of target states. Obviously, the data-driven methods are more sensitive to the changes of observations. They can more timely understand the motion-model and thus more correctly estimate the target states. Yet, the data-driven methods need to be trained by enough data which should cover all cases of tracking. In fact, considering that the parameters in the scenes of target tracking is continuous, it is impossible to build such a dataset filled with "enough data". Hence, problems of generalization insufficiency in data-driven methods are inevitable.

To solve both the problems of model estimation and generalization insufficiency in model-driven and data-driven methods, this paper explores the features of noisy observations and

their relationship to target states based on a digital twin system, which is built for maneuvering-target tracking. Then, two independent networks are designed to construct a complete model estimation for each tracking iteration. One is a noise elimination network (NEN) and the other is a motion-model estimation network (MMEN). The previous one is implemented by a transformer [6] + long short-term memory (LSTM) [7] structure which estimates the actual variation of sequences under the noise interference. The latter one is implemented by a variant of CADP-NN to directly estimate the motion-model according to the observation information after noise elimination. Finally, the estimated motion-model can be use in normal UKF to complete a whole maneuvering-target tracking. The simulation results verify that the proposed tracking method outperforms both the model-driven and data-driven methods, from both the perspectives of tracking accuracy and generalization ability. The main contributions of our work are summarized as follows:

- A digital twin system is designed, and two features in the trajectory created by this system are discussed: one is that the interference cased by noise can be offset with the growth of length of trajectory; the other is that with the growth of length, the shape of trajectory gradually depends on the state transition matrix rather than initial states.
- To effectively eliminate the noise and estimate the actual variation of sequences, a NEN is built by combining the transformer and LSTM together. In NEN, all the variations of noisy sequences and their filtering results under different frequency bands are extracted by transformer structure and the sequential changing information is caught by LSTM structure.
- The motion-model is estimated by a MMEN network which utilizes a convolution neural network to calculate global features of the trajectories and output an estimated transition matrix for maneuvering-target tracking.

The organizations of remainders of this paper are structured as follows. In Related Work Section, two types of maneuvering-target tracking methods are described. In the Exploration of Data Features of Target-Maneuver Based on Digital Twins Section, a digital twin system is established, and two features in the trajectory created by this system are discussed. In Intelligent State Prediction Method Section, the NEN and MMEN are described in detail and a whole tracking process is constructed. In Simulations Section, a low-maneuvering and high-maneuvering scenes are built with different observation noises. Then, our method is utilized to track the target in the two scenes in comparison with classic model-driven and data-driven methods. Finally, the conclusion and future work of the proposed method are given in Conclusion Section.

II. Related Work

In the field of maneuvering-target tracking, the tracking methods are divided into two types: model-driven and data-driven methods. The earliest version of model-driven methods are multiple-model (MM) methods [8], [9], which simply combine different motion-models according to their likelihoods in tracking. To improve the tracking performance, fixed structure MM (FSMM) methods [10], [11] are proposed, which use fixed structures to effectively combine different motion-models. Further, the variable structure MM (VSMM) methods [8], [12] and hybrid grid MM (HGMM) methods [13] are proposed to improve the combination structures of different models to make them more flexible for different maneuvering-tracking scenes. Nowadays, an advance MM method which fuses modified input estimator and best linear unbiased estimation together (MIE-BLUE-IMM) [14] is proposed to further improve the tracking performance. Those MM methods improve the tracking accuracy by given the real-time estimation of the motion-model with different combinations of known models. However, restricted by the accuracy of model presetting and time-delay of estimation calculation [15], the tracking performance of those MM methods are still unsatisfactory for practical applications, especially when targets maneuver with high turn rates.

With the development of deep neural network, a series of data-driven methods [15]–[18] are proposed. The essence of data-driven methods is to construct a complicated end-to-end network to directly predict target states based on the noisy observation data. Under the condition that networks are well-trained in certain datasets, those methods can indeed timely and precisely predict target states with noisy observations. But in fact, all of those observation-to-state networks face a problem of insufficient generalization ability because the target states and observation noises vary too widely to be represented by limited datasets. Once the target states or observation noises are out of the ranges of training data, performances of those data-driven methods will severely degrade. To avoid training the network to adapt to such data with widely varying range, Liu [19] further proposed a cross-and-dot-product neural network (CADP-NN) to learn the motion-models, which vary relatively less. Thus, the prediction performance of CADP-NN becomes more stable when inputs vary out of the training sets. However, the inputs of CADP-NN are target states which cannot be known before tracking. Hence, CADP-NN needs common UKF with predefined motion-models to offer predicted states. Although there is a double-channels method in [19] to maximize the use of estimated motion-models from CADP-NN to improve tracking performance, the actual tracking accuracy is still limited by the UKF itself. Thus, the tracking performance of CADP-NN is still not good enough.

III. The Exploration of Data Features of Target-Maneuver Based on Digital Twins

The key for tracking is to understand the target motion-models, which can be discovered by the data of target states and observations. However, the actual target states and the corresponding observations are very scarce, which are far from meeting the need of motion-model understanding. Although, Liu proposed a LAST database [15] to increase the amount of maneuvering-trajectory data based on the simulations of target tracking of air traffic control (ATC) system [15]. Limited by the simulated parameter settings, the LAST database is

TABLE I
NOTATION LIST.

Notation	Meaning of the notation
$\boldsymbol{f}(\cdot)$	Nonlinear transition function.
$\boldsymbol{h}(\cdot)$	Nonlinear observation function.
k	Discrete time step for tracking, $k = 1, 2, 3, ..., K$.
$\boldsymbol{x}_k$	Target state vector at time step k.
$\boldsymbol{n}_k$	Transition noise at time step k.
$\boldsymbol{z}_k$	Observation vector at time step k.
$\boldsymbol{m}_k$	Observation noise at time step k.
$\boldsymbol{F}$	State transition matrix.
$p_{x,k}, p_{y,k}$	Positions in X and Y-coordinates at time step k.
$v_{x,k}, v_{y,k}$	Velocities in X and Y-coordinates at time step k.
α	Turn rate of maneuvering target.
s_τ	Sampling interval.
n_p, n_v	Transition noises of position and velocity.
σ_p, σ_v, σ_a, σ_θ, σ_d	Standard deviations of position, velocity, acceleration, azimuth and distance, respectively.
θ_k, d_k	Azimuth and distance observed at time step k.
m_θ, m_d	Observation noises of azimuth and distance.
$BW(\boldsymbol{pn}, c)$	Butter-worth filter with one-dimension positions sequence $\boldsymbol{pn}$ and the ratio of cutoff frequency c.
$\boldsymbol{r}_{NEN,k}$	The kth time slice of target matrix to train NEN.
$\widetilde{\boldsymbol{r}}_{NEN,k}$	The kth time slice of the output of NEN.
$\Delta p_{x,k}, \Delta p_{y,k}$	Position bias data in X and Y-coordinates at time step k.
v_k^{sin}, v_k^{cos}	Sine and cosine of intersection angel.
TS	Number of time steps.
SN	Dimension of State vector.
C	Number of channels in feature extractor cell.
$\boldsymbol{CR}$	Output of feature extractor cell.
$\boldsymbol{Z}, \Sigma$	Center and bias of the Gaussion activation in feature extractor cell.

only suitable for common maneuvering-target tracking scenes which cover the distances of target from radar: 0.5∼20 nautical miles; velocities of targets: 0∼340 m/s; maneuvering turn rates: -10∼10 °/s [15]. Obviously, these data do not cover all the cases of target-maneuvers, because targets can move in all the spatial scopes with all kinds of velocities and turn rates satisfying the physical constraints. Worse still, the parameters in motion-models are continuous. Hence, a complete database that covers all cases of target-maneuvers are impossible. Inevitably, training on these incomplete databases, the generalization performance of network will be greatly affected.

A. Establishment of digital twin system

Hence, to exactly understand the target motion-model, this paper built a digital twin system (DTS) for maneuvering-target tracking simulation (MTTS). Based on this DTS, the data features of target-maneuver are explored, and the motion-model can be further solved. All the notations used in this paper are listed in Table I.

Generally speaking, the relationship of target states and observations is represented by the state space model (SSM) as follows ,

$$\textbf{Transition equation}: \quad \boldsymbol{x}_k = \boldsymbol{f}(\boldsymbol{x}_{k-1}, \boldsymbol{n}_k), \tag{1a}$$

$$\textbf{Observation equation}: \quad \boldsymbol{z}_k = \boldsymbol{h}(\boldsymbol{x}_k, \boldsymbol{m}_k), \tag{1b}$$

where $\boldsymbol{f}(\cdot)$ and $\boldsymbol{h}(\cdot)$ are the nonlinear transition function and observation functions, respectively. In (1a), $\boldsymbol{x}_{k-1}$ and $\boldsymbol{n}_k$ are the inputs of transition equation, which denote the target state at time step $k-1$ and transition noise at time step k, respectively. The output of (1a) is the target state at time step k. Hence, (1a) describes the movement of target along with time changing, which is the motion-model and can be used to build the target trajectories in our DTS. But in fact, these trajectories cannot be directly obtained, whose information can only be discovered by observing. (1b) describes this process. That is, at time step k, we only know the observation $\boldsymbol{z}_k$ with the observation function whose inputs are target state $\boldsymbol{x}_k$ and observation noise $\boldsymbol{m}_k$, respectively. Commonly, the transition and observation noises are only random additive noise, which have no relationship with times. Further, for most of the cases in target tracking, the motion-model (1a) can be simulated as a linear model [20]. Hence, (1) can be simplified as follows,

$$\textbf{Transition equation}: \quad \boldsymbol{x}_k = \boldsymbol{F}\boldsymbol{x}_{k-1} + \boldsymbol{n}, \tag{2a}$$

$$\textbf{Observation equation}: \quad \boldsymbol{z}_k = \boldsymbol{h}(\boldsymbol{x}_k) + \boldsymbol{m}, \tag{2b}$$

where $\boldsymbol{F}$ is the state transition matrix which determines the motion-mode of targets. In this paper, we only consider the tracking problem in X-Y plane coordinate. Thus, $\boldsymbol{x}_k$ is defined as $[p_{x,k}, \quad p_{y,k}, \quad v_{x,k}, \quad v_{y,k}]^{\mathrm{T}}$, where $[p_{x,k}, \quad p_{y,k}]^{\mathrm{T}}$ is the two-dimensional (2-D) position, and $[v_{x,k}, \quad v_{y,k}]^{\mathrm{T}}$ is the corresponding velocity.

Generally speaking, for most 2-D maneuver cases, the motion-models are naturally constant turn (CT) models [20], which can be defined as

$$\boldsymbol{F} = \begin{bmatrix} 1 & 0 & \frac{\sin(\alpha s_\tau)}{\alpha} & \frac{\cos(\alpha s_\tau)-1}{\alpha} \\ 0 & 1 & \frac{1-\cos(\alpha s_\tau)}{\alpha} & \frac{\sin(\alpha s_\tau)}{\alpha} \\ 0 & 0 & \cos(\alpha s_\tau) & -\sin(\alpha s_\tau) \\ 0 & 0 & \sin(\alpha s_\tau) & \cos(\alpha s_\tau) \end{bmatrix}. \tag{3}$$

In (3), s_τ is the sampling interval of trajectories, α is the turn rate of maneuvering target. Obviously, when α is 0, $\boldsymbol{F}$ degenerates into the following form:

$$\boldsymbol{F} = \begin{bmatrix} 1 & 0 & s_\tau & 0 \\ 0 & 1 & 0 & s_\tau \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{4}$$

which is called constant velocity (CV) model. The transition noise $\boldsymbol{n}$ can be defined in the form of normal distribution as

$$\begin{aligned} &\boldsymbol{n} = [n_p, n_p, n_v, n_v]^{\mathrm{T}}, \\ &n_p \sim \mathcal{N}(n_p; 0, \sigma_p^2), n_v \sim \mathcal{N}(n_v; 0, \sigma_v^2), \end{aligned} \tag{5}$$

where $\sigma_p = 0.5\sigma_a s_\tau^2$ and $\sigma_v = \sigma_a s_\tau$ are the standard deviations of transition noise for distance and velocity, respectively; σ_a is the standard deviation of accelerated velocity noise. Moreover, for common radar tracking system, the observation equation is defined as:

$$\underbrace{\begin{bmatrix} \theta_k \\ d_k \end{bmatrix}}_{\boldsymbol{z}_k} = \begin{bmatrix} \arctan\dfrac{p_{y,k}}{p_{x,k}} \\ \sqrt{p_{x,k}^2 + p_{y,k}^2} \end{bmatrix} + \underbrace{\begin{bmatrix} m_\theta \\ m_d \end{bmatrix}}_{\boldsymbol{m}}. \tag{6}$$

where $\boldsymbol{m} = [m_\theta, \quad m_d]^{\mathrm{T}}$ is the noise of observation vector. It contains the azimuth and distance parts, i.e., m_θ and m_d,

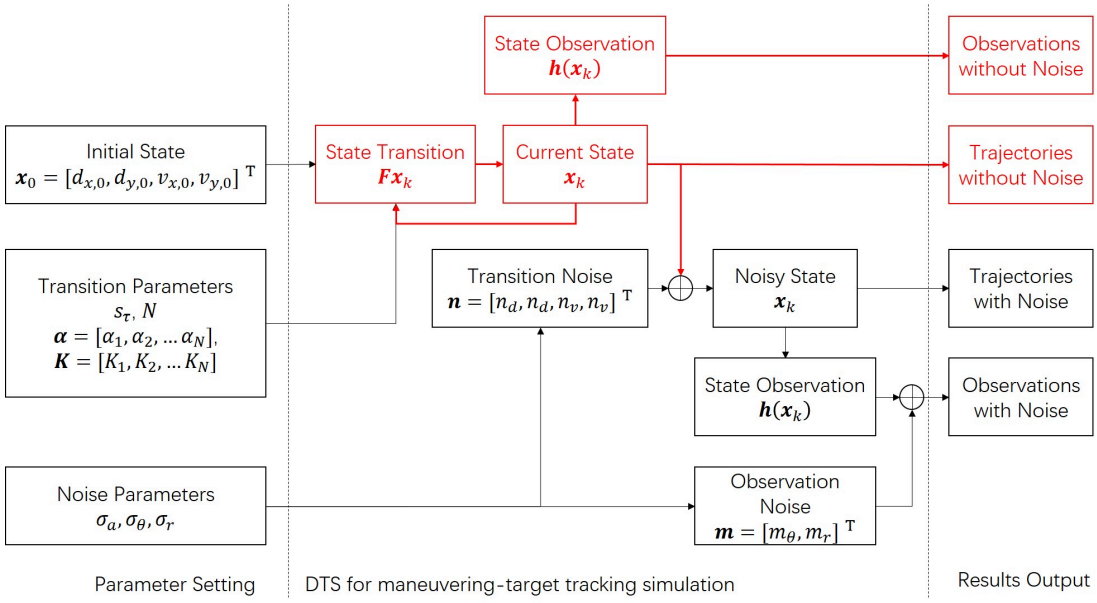


Fig. 1. DTS for MTTS.

which are also defined in the form of normal distribution as,

$$m_\theta \sim \mathcal{N}(m_\theta; 0, \sigma_\theta^2), \quad m_d \sim \mathcal{N}(m_d; 0, \sigma_d^2), \tag{7}$$

where σ_θ and σ_d are the standard deviations of azimuth and distance, respectively. Based on the equations from (2) to (7), the DTS is built in Figure 1.

For each special tracking scene, given a predetermined initial state $\boldsymbol{x}_0$ and sampling interval s_τ, a trajectory and corresponding observations without noises are generated in the red part. Each trajectory is combined by N segments. In the nth segment, the motion-model and duration are determined by turn rate α_n and sampling times K_n. Further, the noises can be added to the calculation process to simulate the real tracking scene with parameters σ_a, σ_θ and σ_d. Then, the trajectory and corresponding observations with noises are generated in the black part.

Different from traditional dataset, our DTS can generate all kinds of maneuvering trajectories, because the parameters for trajectory generation with DTS are unlimited. Figure 2 shows two maneuvering trajectories and their observation data. One contains 3 segments with turn rates: -3, 8, 0 $^\circ/s$ and numbers of time steps in each segment: 300, 300, 300. The other contains 5 segments with turn rates: 10, -5, 6, -2, 8 $^\circ/s$ and numbers of time steps in each segment: 200, 200, 200, 200, 200. The observations of two trajectories also show different features with different noise deviations. Obviously, we can use all kinds of parameters to generate trajectories and their observations to adapt to various application needs. There are two advantages of our DTS:

1 Data with and without noise are generated in pairs, which can help us to analyze the interference of noise on data.
2 Trajectories and observations are generated in pairs, which can help us to explore the information of trajectories from observations.

B. Exploration of data features based on DTS

Based on the DTS, we investigate the interference of noise for tracking, according to the comparison between data with and without noise.

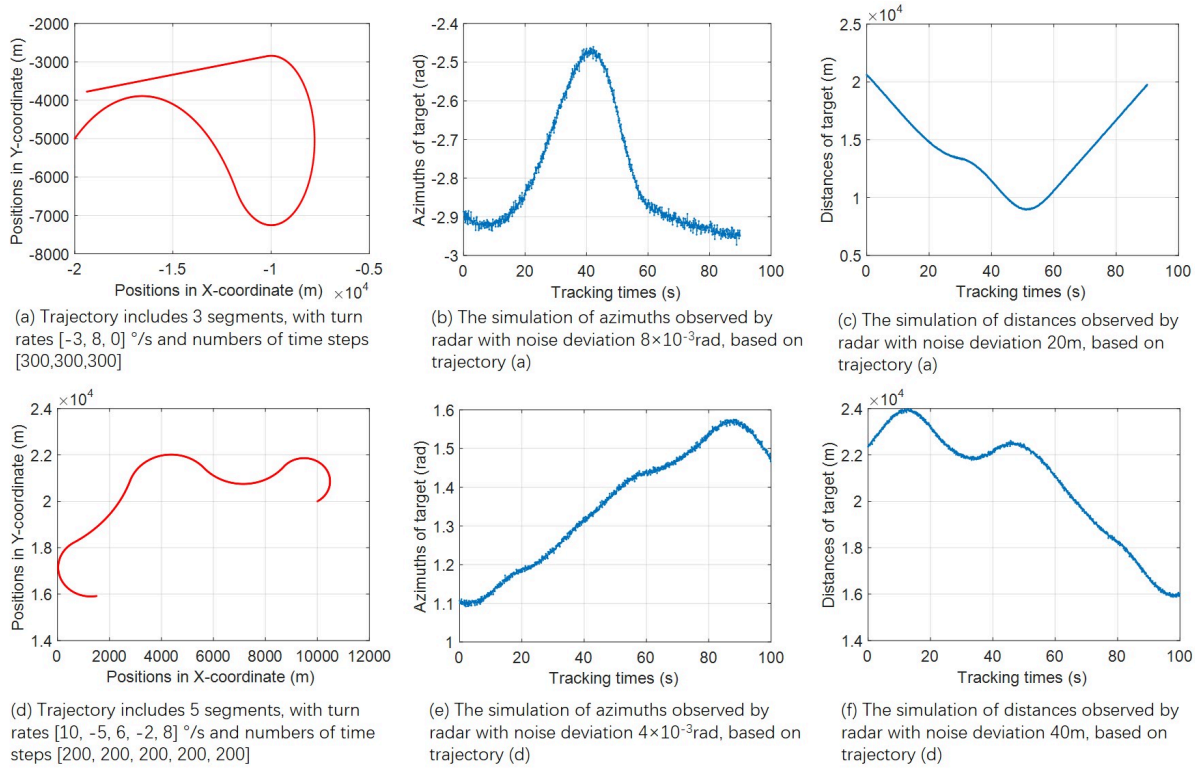
Observation 1: The interference cased by noise can be offset by the growth of length of trajectory .

Analysis 1: We increase the deviations σ_θ and σ_d to analyze the interference of observation noises on trajectory recognition in Figure 3. In each sub-figure, the black line is the original trajectory, and the red line is the noisy one, which is calculated by noisy observations as follows,

$$p_{x,k} = d_k cos(\theta_k), \tag{8a}$$
$$p_{y,k} = d_k sin(\theta_k). \tag{8b}$$

The captions of each sub-figure contain a set of parameters which are used in DTS to generate the corresponding trajectory and observations. In the sub-figure (a), the parameters are set as: $K = 50$, $\sigma_\theta = 4 \times 10^{-3} rad$ and $\sigma_d = 10m$. Keeping the tracking time steps K unchanged, we increase the noisy deviations to 3 and 9 times in sub-figure (b) and (c), respectively. Obviously, the noisy trajectory becomes more and more difficult to recognize along with the increasing of noise deviation. But fortunately, the observation noise is stationary. Based on DTS, we further extend the noisy trajectory sequences to their 3, 9 and 27 time lengths, which are shown in sub-figures (f), (e) and (d), respectively. Their down-sampling ones, which are all down-sampling to time steps 50, are shown in sub-figures (i), (h) and (g), respectively. From those down-sampling sub-figures we can see that the interference of noise declines when sequence length increases. Specifically, comparing figures (a) and (g) we can see that the interference of noise is similar. Hence, the increased effect cased by increased noise deviation has been offset by increased sequence length.

This completes the analysis of Observation 1.



(a) Trajectory includes 3 segments, with turn rates [-3, 8, 0] °/s and numbers of time steps [300,300,300]

(b) The simulation of azimuths observed by radar with noise deviation 8×10^{-3}rad, based on trajectory (a)

(c) The simulation of distances observed by radar with noise deviation 20m, based on trajectory (a)

(d) Trajectory includes 5 segments, with turn rates [10, -5, 6, -2, 8] °/s and numbers of time steps [200, 200, 200, 200, 200]

(e) The simulation of azimuths observed by radar with noise deviation 4×10^{-3}rad, based on trajectory (d)

(f) The simulation of distances observed by radar with noise deviation 40m, based on trajectory (d)

Fig. 2. Samples of two maneuvering trajectories data.

According to observation 1, the interference of noise is relative. If the interference can be eliminated under certain noise deviations, the ones with larger deviations can also be eliminated with longer trajectories.

Moreover, according to the relationship between state and observation shown in (6), we can find that there is in fact no velocity information in observations. Hence, it is impossible to deduce the target state from observation based on this actual relationship. That is those end-to-end networks which predict target states by observations are not in fact based on this actual relationship. In other words, the trajectories are not deduced based on actual relationship in (2). In most of the cases, those networks are trained to memory the patterns of trajectories according to observations. If the observations do not exist in training, the network cannot find the correct trajectories. Fortunately, based on DTS, the observation sequences can be completely generated to further explore their features.

Observation 2: With the growth of length, the shape of trajectory gradually depends on the state transition matrix rather than initial states.

Analysis 2: Equation (2) tell us the fact that main influence factors in trajectory generation are the initial state and state transition matrix $\boldsymbol{F}$. We test the influence of two factors when biases happen. Figure 4 shows the changes of trajectories when biases of transition matrix F and initial state $\boldsymbol{x}_0$ happen. In original trajectory, the initial state $\boldsymbol{x}_0$ is set to be $[2000, 2000, 200, 200]^{\mathrm{T}}$, the turn rate in F is set to be 5° and the time step is set to be 200. We can find this original trajectory with mark of blue circle in Figure 4. The 'F-biased' one, marked with red star, is the trajectory with turn rate in F changed to 7° and the '$\boldsymbol{x}_0$-biased' one, marked with black upper-triangle, is the trajectory with initial state $\boldsymbol{x}_0$ changed to be $[2100, 2100, 210, 210]^{\mathrm{T}}$. Obviously, '$F$-biased' one deviates from the original trajectory more than the '$\boldsymbol{x}_0$-biased' one along with trajectory lengthening.

Moreover, we investigate the deviations of the two biased trajectories from original trajectory in Figure 5. In this figure, both deviation values increase along with the numbers in positions sequence. Obviously, the deviation increment of 'F-biased' is exponential but the one of '$\boldsymbol{x}_0$-biased' is linear. Hence, an incorrect transition matrix will lead to much larger errors in trajectory generation than incorrect initial state along with trajectory lengthening. In other words, the precision of trajectory generation gradually depends on the state transition matrix rather than initial states when the length of trajectory increase.

This completes the analysis of Observation 2.

Observation 2 shows that, under a proper trajectory length, we can exactly tell the transition matrix without the information of initial state.

IV. Intelligent State Prediction Method

Based on data features explored from DTS, we know that the interference of noise on sequence is under control and the transition matrix can be derived under proper trajectory length. Hence, we can actually understand motion-model of maneuvering target by minimizing the interference of noise on observations. To this end, a noise elimination network (NEN) is constructed to minimize the noise interference, and a motion-model estimation network (MMEN) is also constructed

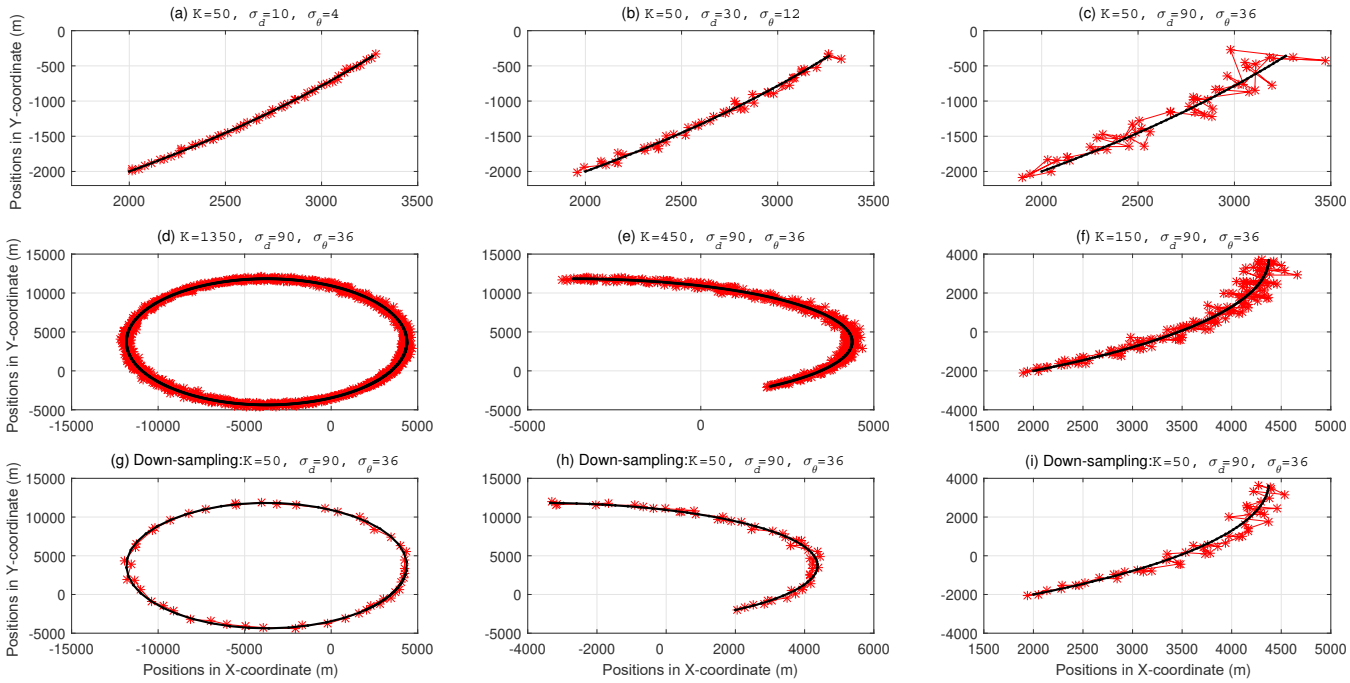


Fig. 3. The effects of noise on trajectories.

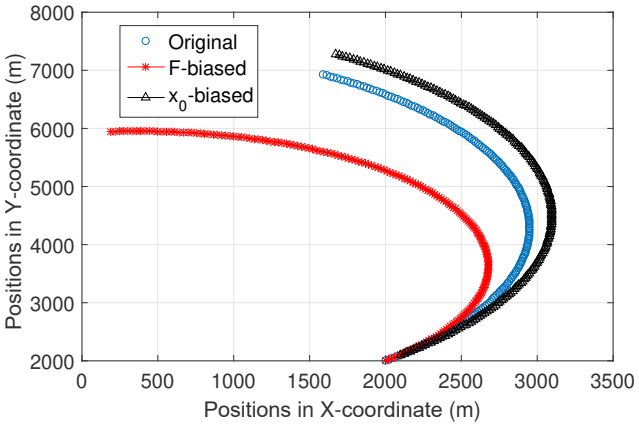


Fig. 4. Trajectories comparison when biases happen.

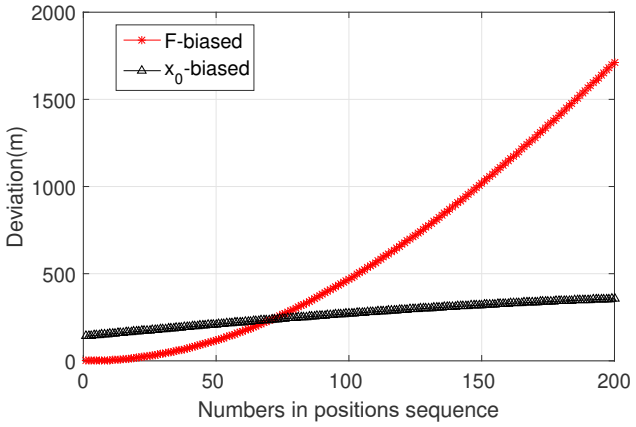


Fig. 5. Deviations comparison between two biased trajectories.

to correctly estimate the motion-model given noise-eliminated observations. Then we combine the two networks together to construct our intelligent state prediction method (ISPM).

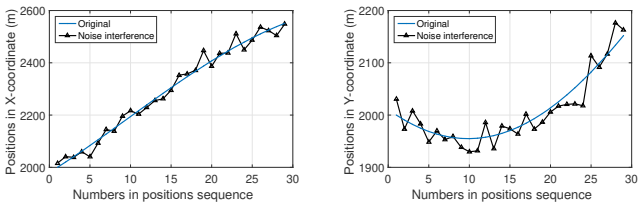
A. Noise elimination network

Although we can reduce the effect of noise on trajectory by extending the trajectory length, a noise eliminating method is still needed to further eliminate the noise interference in case of insufficient length.

1) Problems of noise interference: Considering the need for real-time tracking, the lengths of trajectories used to predict the motion-models can be limited to 29. In that case, we set the parameters in DTS in Table II. Then we investigate the interference of noise on trajectory recovered from noisy observations with (8). Figure 6 shows the effects of noise interference on position data in X and Y-coordinates in this trajectory. In Figure 6, the initial state and turn rate are set to be $[2000, 2000, 200, 200]^{\mathrm{T}}$ and $30°/s$. Under such a common scene, the interference of noise on trajectory recovery is serious, which will lead to a bad recognition of trajectory

TABLE II
PARAMETER SETTING FOR NOISE ELIMINATION.

Names	Notations	Values
Deviation of accelerated velocity noise	σ_a	$10m/s^2$
Deviation of azimuth noise	σ_θ	$8\times10^{-3}rad$
Deviation of distance noise	σ_d	$20m$
Number of time steps	K	29
Sampling interval	s_τ	0.1s



(a) Position data in X-coordinate (b) Position data in Y-coordinate

Fig. 6. Noise interference on position data in X and Y-coordinates.

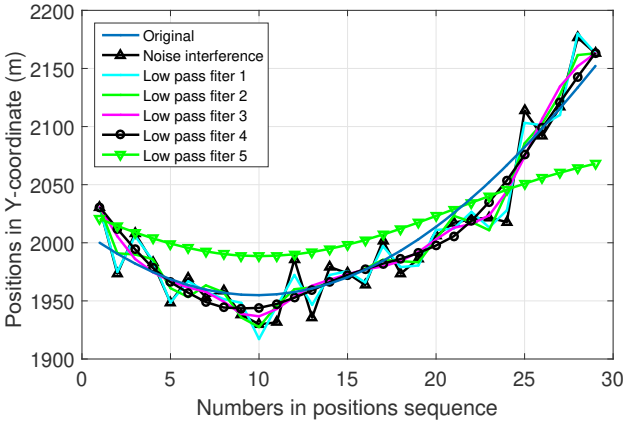


Fig. 7. Filtering the trajectory with low-pass filters.

from observations.

2) Network structure for noise elimination: Considering the fact that the main information of trajectory is in low frequencies, a set of butter-worth filters are designed to eliminate the noise. The filtering results of Y-position data are shown in Figure 7. In this figure, the ratios of cutoff frequencies from filters 1 to 5 are set to be 0.8, 0.6, 0.4, 0.2 and 0.05, respectively. As seen in Figure 7, the filtering results become smoother with the ratios going down. Among of those results, the filter 4 is closest to the original Y-position. However, all filtering results are biased including filter 4 in such a short time range. To improve the performance of filtering, this paper builds a NEN with transformer-LSTM structure as shown in Figure 8. Transformer which contains encoder and decoder parts can search the key information and effectively understands the input itself according to the self-attention network structure [6]. This paper utilizes the transformer to explore the key information in both observations and filtering results, and recovers the noiseless trajectory as accurate as possible. Moreover, considering the fact that those filtering results are only associated with the information of frequency, we add the LSTM [7], [21], [22] structure to explore the temporal relationships between states.

Specifically, the NEN is only designed to eliminate the noises of one-dimensional sequence. Taking positions sequence in Y-coordinate as an example, we define this sequence as $\boldsymbol{pn}$. Then we compose the input matrix with the filtering results of $\boldsymbol{pn}$ as $[\boldsymbol{pn}, BW(\boldsymbol{pn}, 0.8), BW(\boldsymbol{pn}, 0.6), BW(\boldsymbol{pn}, 0.4), BW(\boldsymbol{pn}, 0.2), BW(\boldsymbol{pn}_y, 0.05)]$, where $BW(\boldsymbol{pn}, c)$ is the butter-worth filer with the ratio of cutoff frequency c. Then, the matrix is fed into a Transformer-Encoder to get a memory matrix with shape $TS \times SN$, where TS is the number of time steps, and it equals 29 in our paper, SN is the state number, and it equals 6. Then, the memory is fed into a bidirectional LSTM layer, named LSTM_encoder_layer, to obtain an information encoder sequence whose shape is $TS \times 2SN$. This sequence is fed into a Transformer-Decoder combining with the target sequence which is calculated by a Linear_encoder_layer. The output of Transformer-Decoder will be further fed into another LSTM layer, named LSTM_decoder_layer to calculate the decoder information of noise elimination. Then, we add this decoder information to the input to eliminate noise and feed it into another linear layer to get the final output. The output is separated into two parts: one is the noise-limited positions sequence in Y-coordinate and the other is the original filtering results. The essence of NEN is that we combine both information of noisy sequence and its filtering data together to reconstruct the noise-limited one, instead of directly mapping the noisy data itself to noiseless one. Hence, our MEN learned a more stable and simple data relationship than common direct mapping, which guarantees the strong generalization ability of our NEN.

3) The training of NEN: The loss is defined as the Mean Square Error (MSE) loss:

$$\mathcal{L}_{NEN} = \frac{1}{K}\sum_{k=1}^{K}(\widetilde{\boldsymbol{r}}_{NEN,k} - \boldsymbol{r}_{NEN,k})^2, \tag{9}$$

where $\widetilde{\boldsymbol{r}}_{NEN,k}$ is the kth time slice in Output, and $\boldsymbol{r}_{NEN,k}$ is the kth time slice in target matrix which is defined as $[\boldsymbol{p}_y, BW(\boldsymbol{pn}, 0.8), BW(\boldsymbol{pn}, 0.6), BW(\boldsymbol{pn}, 0.4), BW(\boldsymbol{pn}, 0.2), BW(\boldsymbol{pn}, 0.05)]$. $\boldsymbol{p}_y$ is the noiseless positions sequence in Y-coordinate. We train the NEN on data derived from DTS with parameters shown in Table II, and other parameters are set randomly from the range: $d_0 \in [1000, 10000m]$; $v_0 \in [50, 350m/s]$; $\alpha \in [-90, 90°]$, where $d_0 = \sqrt{p_{x,0}^2 + p_{y,0}^2}$, $v_0 = \sqrt{v_{x,0}^2 + v_{y,0}^2}$.

In training process, we found that the network converges quickly in the first 1000 training rounds. After 1000 training rounds, the downward trends of losses are no longer significant. The details can be found in Figure 9. Even so, a long time training can be helpful for noise elimination performance of NEN based on existing experimental results. Moreover, there is no need to strictly generate our training data based on Table II. For example, we can generate the training data with parameter $\alpha \in [0, 90°]$. Then, when our NEN converges, it can also be suitable for those data with parameter $\alpha \in [-90°, 0]$.

After training, we can obtain the noise-limited sequence as shown in Figure 10. The original positions sequence in Y-coordinate is shown with blue line, the noisy one is shown with upper triangle black line, the noise-limited one is shown with star red line, and other sequences after different filtering can be found in this figure. Obviously, the noise-limited sequence is mainly reconstructed by the information of filtering results

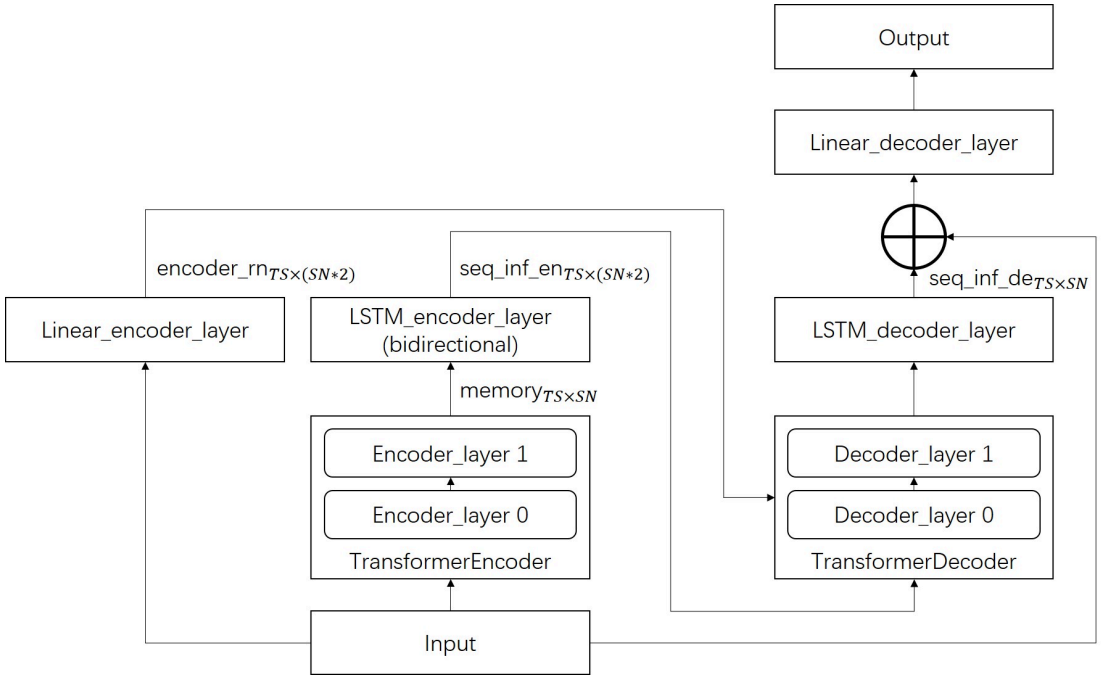


Fig. 8. NEN structure.

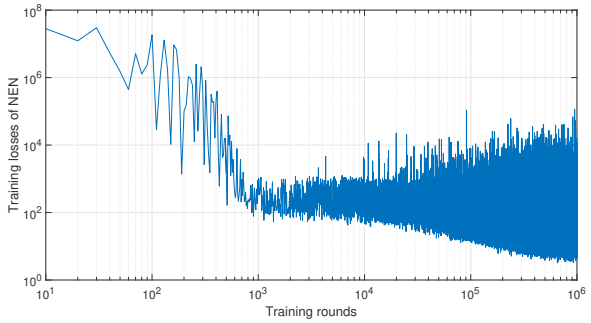


Fig. 9. The convergence of NEN in training.

of filter 4. But at the same time, other information of filtering results are considered to minimum this biased errors, which can be further verified in Figure 11. In this figure we can see that the errors of noise-limited results is most similar to the ones of filter 4, but it is also affected by other filters to become smoother. Hence, noise-limited result can well restore the information of original sequence. Moreover, another NEN can also be defined to eliminate the noise in X-position data sequence. Then, those two noise-limited results can be combined to form a noise-limited trajectory.

B. Motion-Model Estimation Network

According to the tracking process, the key to track the maneuvering-target is to timely and correctly estimate the motion-model. Based on the DTS, we know that the motion-model can be represented by state transition matrix $\boldsymbol{F}$. In other words, we need to timely and correctly estimate the $\boldsymbol{F}$ to guarantee the accuracy of tracking. In (2), we know that $\boldsymbol{F}$ is related to target states. This relationship has been deduced in [19] by the CADP-NN. But in fact, target states contain both position and velocity information, yet observation only contains the information of position without velocity. Hence, this paper proposes a MMEN to solve this problem.

1) Problems of motion-model estimation: Obviously, different from CADP-NN, we have to estimate the motion-model only by observations which contain no velocity information. Fortunately, according to Observation 2, the shape of trajectory mainly depends on $\boldsymbol{F}$, not initial states. We can use the information of the shape of trajectory, which is in fact the positions sequence and bijective to observations, to estimate $\boldsymbol{F}$.

To better utilize this information, this paper proposed a hierarchical-difference matrix T^{hd} to explore the shape information as follows,

$$T^{hd} = [[p_{x,0}, p_{y,0}]^{\mathrm{T}}, [p_{x,1}, p_{y,1}]^{\mathrm{T}}, ...[p_{x,K}, p_{y,K}]^{\mathrm{T}}] - [p_{x,0}, p_{y,0}]^{\mathrm{T}}, \quad (10)$$

where the subtrahend $[p_{x,0}, p_{y,0}]^{\mathrm{T}}$ is broadcast to the shape of minuend, i.e., the positions in X and Y-coordinates. Further, to completely focus on the shape of trajectory, we only consider the information of intersection angle between the vectors of T^{hd}. Hence, the values of sine and cosine of intersection angle are calculated by cross and dot product (CADP) as shown in following,

$$\textbf{Sine}: \quad v_k^{sin} = \frac{\Delta p_{x,k}\Delta p_{y,k+1} - \Delta p_{y,k}\Delta p_{x,k+1}}{|\Delta \boldsymbol{p}_k||\Delta \boldsymbol{p}_{k+1}|}, \quad (11a)$$

$$\textbf{Cosine}: \quad v_k^{cos} = \frac{\Delta p_{x,k}\Delta p_{x,k+1} + \Delta p_{y,k}\Delta p_{y,k+1}}{|\Delta \boldsymbol{p}_k||\Delta \boldsymbol{p}_{k+1}|}, \quad (11b)$$

where $\Delta p_{x,k} = p_{x,k} - p_{x,0}$, $\Delta p_{y,k} = p_{y,k} - p_{y,0}$ $\Delta \boldsymbol{p}_k = [\Delta p_{x,k}, \Delta p_{y,k}]^{\mathrm{T}}$. For each K time steps trajectory, we can get $K-2$ time steps $\boldsymbol{v}^{sin}$ and $\boldsymbol{v}^{cos}$ sequences, which are combined

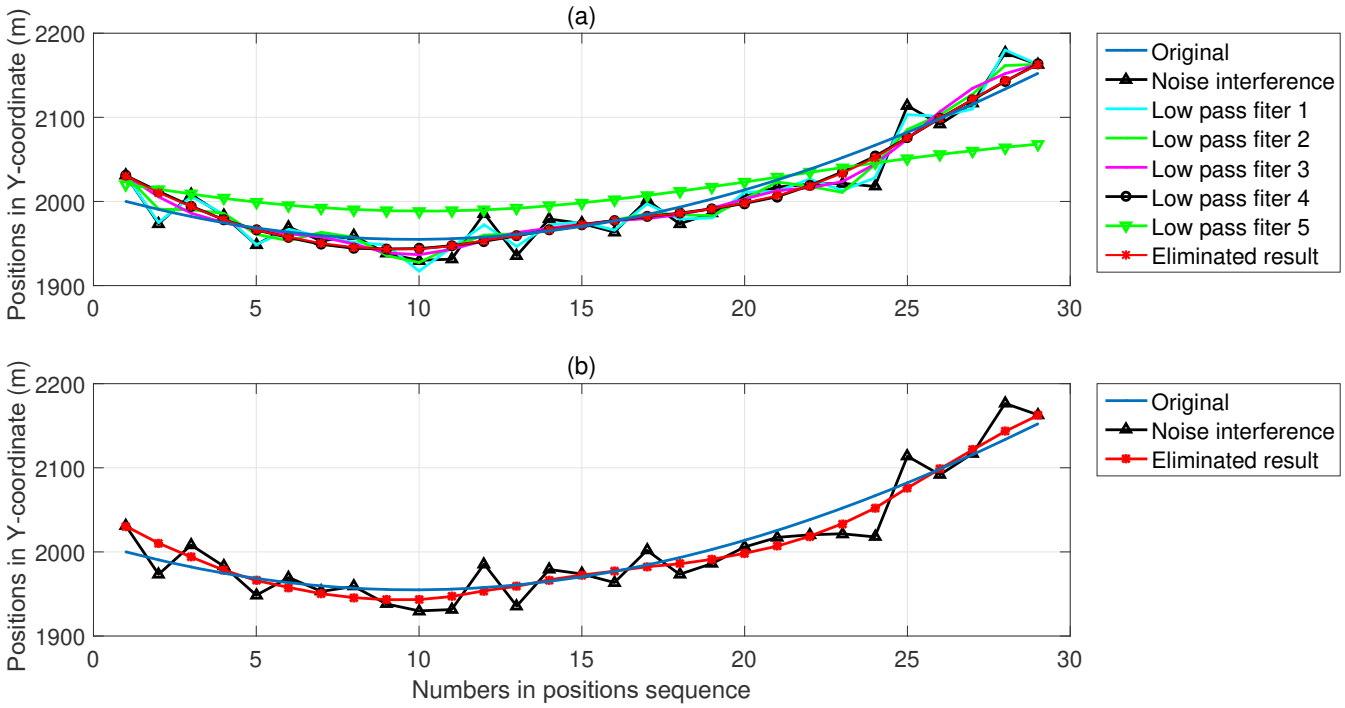


Fig. 10. Noise elimination results.

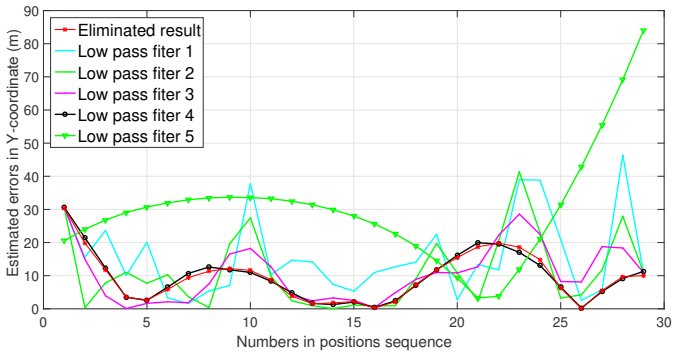


Fig. 11. Noise elimination errors.

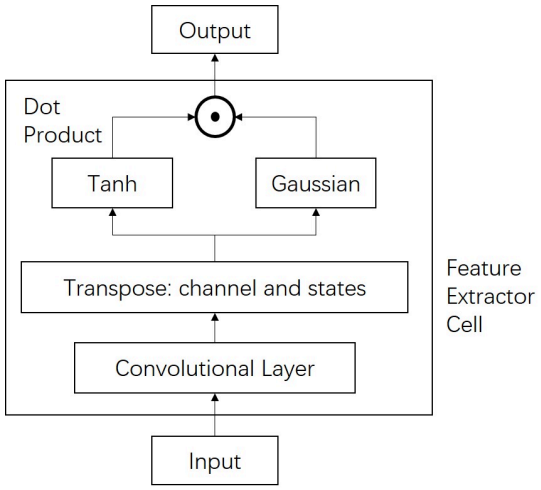


Fig. 12. Feature extractor cell.

to form the input of our MMEN.

2) Structure of MMEN: From the perspective of the shape of positions sequence, we mainly consider the global features in sequence, not the temporal relationship between states. Hence, we utilize the convolutional neural network (CNN) [23] structure to explore these global features. Then, a Gaussian activation is designed to promote the dimension of features for a better shape-features classification. To this end, a feature extractor cell is designed and shown in Figure 12. Assuming the shape of input is $TS \times SN$, we need to expand it to $1 \times TS \times SN$ to meet the need for convolutional layer, in which 1 is the channel number. The convolutional layer contains C kernels whose sizes are all $SN \times SN$, and the stride and padding are set to be 1 and 0 in this layer, respectively. Hence, after through the convolutional layer, the shape of output data, denoted as $\boldsymbol{CR}$, becomes $C \times TS' \times 1$, where $TS' = TS - SN + 1$. Then, the $\boldsymbol{CR}$ is reshaped to be $1 \times TS' \times C$.

Further, we calculate the higher dimensional data by Gaussian activation with learnable parameters: center $\boldsymbol{Z} = \{z_{k,c}\}_{k=1,c=1}^{TS',C}$ and bias $\boldsymbol{\Sigma} = \{\sigma_{k,c}\}_{k=1,c=1}^{TS',C}$. Then we also calculate the signs of each element in CR by tanh activation. Finally, we multiply the Gaussian activation results by tanh activation results as follows,

$$o_{k,c} = \exp[-(\frac{cr_{k,c} - z_{k,c}}{\sigma_{k,c}})^2] \cdot \tanh(cr_{k,c}), \qquad (12)$$

where $cr_{k,c} \in \boldsymbol{CR}$ and $o_{k,c} \in \boldsymbol{O}$. $\boldsymbol{O}$ is the output matrix with shape $1 \times TS' \times C$.

Further, we use four feature extractor cells to construct our MMEN as shown in Figure 13. Specifically, the positions

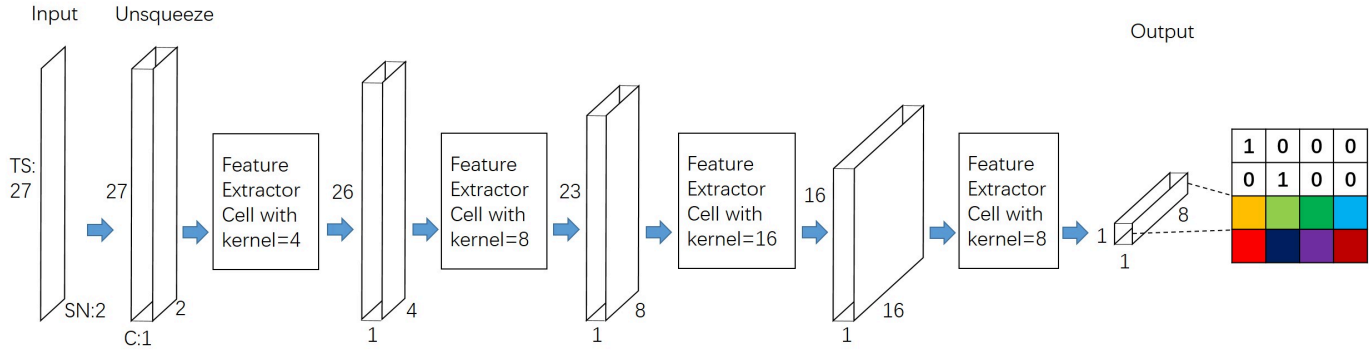


Fig. 13. Motion-model estimation network.

sequences in X and Y-coordinates with time steps 29 are used to estimate the real transition matrix $\boldsymbol{F}$ in this paper. As mentioned before, the CADP is utilized to calculate $\boldsymbol{v}^{sin}$ and $\boldsymbol{v}^{cos}$ sequences which form the input of MMEN with shape 27×2. Then the CADP result is added a dimension for channel number with value 1. The result further goes through four feature extractor cells with kernel numbers 4, 8, 16 and 8 respectively. The output with shape $1 \times 1 \times 8$ is reshaped into 2×4, which is combined with vector [1,0,0,0] and [0,1,0,0] to form the estimated transition matrix $\tilde{\boldsymbol{F}}$.

3) The training of MMEN: The loss used to train MMEN is also the MSE loss shown as follows,

$$\mathcal{L}_{MMEN} = \frac{1}{8}\sum_{j=1}^{4}\sum_{i=3}^{4}(\tilde{\boldsymbol{F}}_{i,j} - \boldsymbol{F}_{i,j})^2. \tag{13}$$

There are two steps for training the MMEN. First, we use noiseless positions sequence in trajectory and the real transition matrix $\boldsymbol{F}$ as the input-output pairs for training. The parameters used to generate the trajectory are randomly sampled from the range: $d_0 \in [1000, 10000m]$; $v_0 \in [50, 350m/s]$; $\alpha \in [-90, 90°]$, which are the same as those set in The training of NEN Subsection. After about 10^5 rounds for training, the loss can decrease to 10^{-8}, as shown in Figure 14. Then,

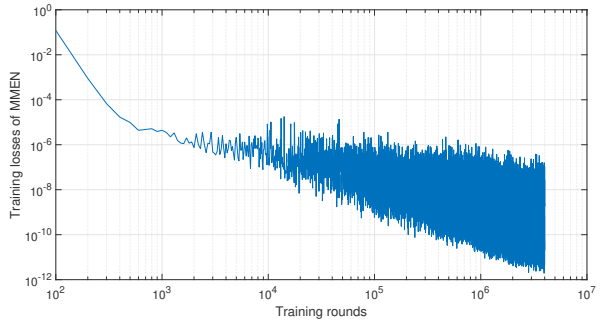


Fig. 14. MMEN training with noiseless input data.

the trained NEN network is used to eliminate the noise for real positions sequence which are calculated directly from observations. Those noise-limited data are fed into MMEN and train MMEN again. The training process is shown in Figure 15. Obviously, after about 10^4 rounds for training, the loss of

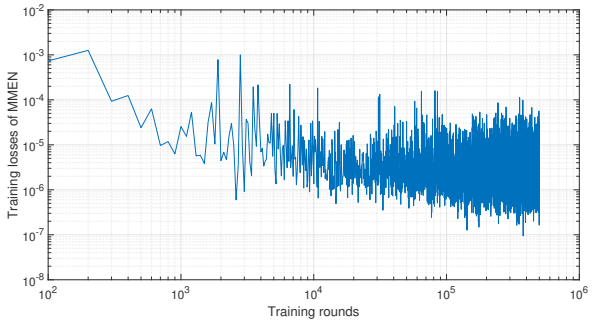


Fig. 15. MMEN training with noise-eliminated input data.

MMEN tends to stabilize around the value of 10^{-6}, which is in fact good enough for practical tracking process.

C. Intelligent state prediction method for tracking

In this section, our ISPM is constructed by combining NEN and MMEN to timely estimate the transition matrix according to noisy observations[1]. Then, the estimated transition matrix can be used in common UKF to predict target states. Specifically, as shown in Figure 16, the noisy observations are timely accumulated. Once the sequence number reaches 29, they will be exacted as a segment, and recovered to be positions sequences in X and Y-coordinates by (8). Then, the two positions sequences are separated and fed into different NEN to eliminate the noise. Next, we combine the noise-limited sequences together and calculate the corresponding $\boldsymbol{v}^{sin}$ and $\boldsymbol{v}^{cos}$ sequences by CADP. Further, those results are fed into MMEN to timely predict $\tilde{\boldsymbol{F}}$. Finally, the state of maneuvering target can be correctly predicted by UKF given this predicted $\tilde{\boldsymbol{F}}$. The tracking process is summarized in Algorithm 1.

V. Simulations

In this section, simulation results are presented to verify the performance of our ISPM. To this end, two tracking scenes are designed, which are the low-maneuvering and high-maneuvering motion scenes, respectively. Then, in comparison with the state-of-the-art data-driven methods: DeepMTT

[1]Open resource codes: https://github.com/ljx43031/DTS_for_ISPM.

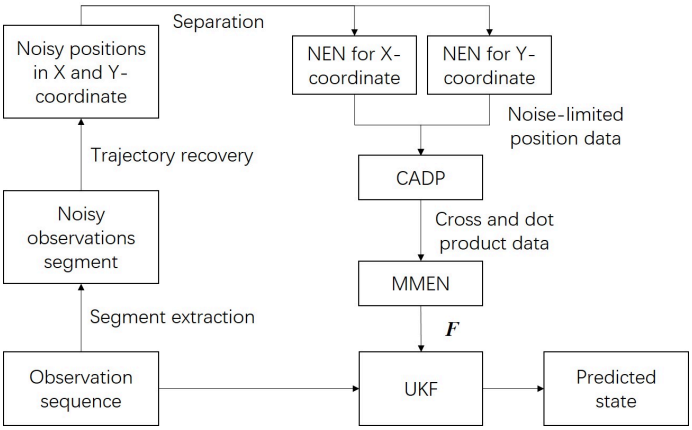


Fig. 16. Intelligent state prediction method.

Algorithm 1: ISPM tracking process.

Input: Noisy observations $\boldsymbol{z}$ with K time steps.
Output: Trajectory $\boldsymbol{x}$ with K time steps.
Initialization: Trained NEN and MMEN; 5 low-pass filters with ratios of cutoff frequencies: 0.8, 0.6, 0.4, 0.2 and 0.05, respectively; $\boldsymbol{x}_0$.
for k *in* K **do**
  while k<=*K-28* **do**
    Segment extraction: $\boldsymbol{z}_{k:k+28}$.
    Position recovery with equation (8), to get noisy positions sequences: $\boldsymbol{p}_{k:k+28}$.
    Noise elimination with 5 low-pass filters: $\boldsymbol{p}_{xne,k:k+28} = NEN(\boldsymbol{p}_{x,k:k+28}, filters)$and $\boldsymbol{p}_{yne,k:k+28} = NEN(\boldsymbol{p}_{y,k:k+28}, filters)$
    Combination of noise-limited results: $\boldsymbol{p}_{ne} = [\boldsymbol{p}_{xne,k:k+28}; \boldsymbol{p}_{yne,k:k+28}]$.
    Transition matrix estimation: $\tilde{\boldsymbol{F}} = MMEN(\boldsymbol{p}_{ne})$.
  end
  Prediction of UKF filter: $\tilde{\boldsymbol{x}}_k = \tilde{\boldsymbol{F}}\boldsymbol{x}_{k-1}$.
  Update of UKF filter: $\tilde{\boldsymbol{x}}_k \xrightarrow{z_k} \boldsymbol{x}_k$.
end

(DMTT) [15] and CADP-NN [19] and model-driven method: MIE-BLUE-IMM [14], we discuss the tracking performance of our ISPM based on the tracking RMSE defined as follows,

$$\boldsymbol{RMSE}_n = \sqrt{\frac{1}{M}\sum_{k=M(n-1)+1}^{Mn}(\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k)^2}, \qquad (14)$$

where $\hat{\boldsymbol{x}}_k$ is the estimated target state, M is the state number in each segment of the whole trajectory. That means, we separate the whole trajectory into N segments. Each segment contains M states. For the nth segment, the $\boldsymbol{RMSE}_n$ is calculated in equation (14). Finally, the $\boldsymbol{RMSE}$ in tracking results of DMTT, CADP-NN and ISPM algorithms are shown and discussed in the following subsection.

A. Simulation scenes

There are two scenes discussed in this paper: the low-maneuvering and high-maneuvering motion scenes. The turn rate in low-maneuvering scene, which contains normal maneuvering cases in ATC system, is usually within the range of -10~10 °/s. In high-maneuvering scene, the turn rate is set to be out of the range of -10~10 °/s. Moreover, as mentioned before, we only consider the X-Y plane coordinate in this paper. Hence, the parameters for trajectory generation in two scenes are set in Table III. Specifically, we designed a trajectory with 3 maneuvering parts for low-maneuvering scene as seen in Figure 17. The duration in each part is 30s, and the turn rates α = are set to be $-3°/s$, $8°/s$ and $0°/s$, respectively. Moreover, the trajectory for high-maneuvering scene is designed with 6 maneuvering parts as seen in Figure 18. In those parts, the sencond, third, and fifth parts are the high-maneuvering part, whose duration is 9s, and turn rates α are $-30°/s$, $15°/s$ and $60°/s$, respectively.

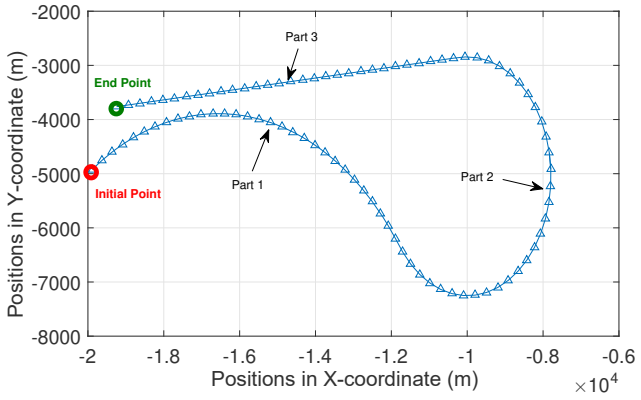


Fig. 17. Trajectory of low-maneuvering.

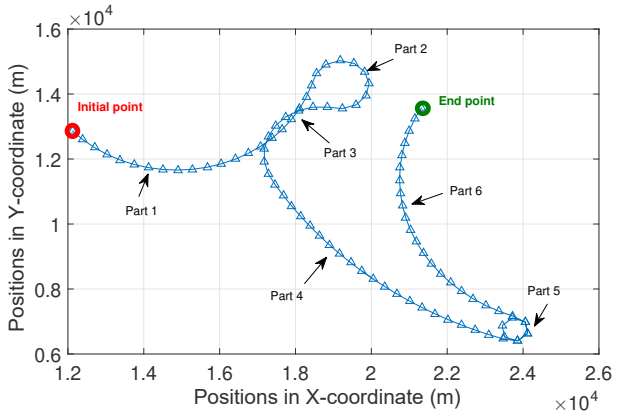


Fig. 18. Trajectory of high-maneuvering.

B. Tracking for low-maneuvering scene

In this section, the tracking results of DMTT, CADP-NN, MIE-BLUE-IMM and ISPM algorithms in low-maneuvering scene are discussed. Obviously, the maneuvering trajectory of this scene is in the range of ATC system. Hence, we

TABLE III
PARAMETER SETTING IN LOW AND HIGH-MANEUVERING SCENES.

Trajectory	Initial State	Part 1	Part 2	Part 3	Part 4	Part 5	Part 6
Low-maneuvering	[-20000m,-5000m, 250m/s,180m/s]	30s, $\alpha = -3°/s$	30s, $\alpha = 8°/s$	30s, $\alpha = 0°/s$	/	/	/
High-maneuvering	[12000m,13000m, 250m/s,-300m/s]	21s, $\alpha = 6°/s$	9s, $\alpha = -30°/s$	9s, $\alpha = 15°/s$	21s, $\alpha = 2°/s$	9s, $\alpha = 60°/s$	21s, $\alpha = -5°/s$

TABLE IV
MEANS OF DISTANCE TRACKING RMSES IN THREE SEGMENTS OF THE TRAJECTORY OF LOW-MANEUVERING SCENE.

Noises	Segments	MIE-BLUE-IMM (m)	DeepMTT (m)	CADP-NN (m)	ISPM (m)
$(2 \times 10^{-3} rad, 4m)$	**The first part**	16.00	36.72	34.82	6.47
	The second part	15.23	32.70	41.24	7.69
	The third part	13.35	20.09	9.43	12.60
$(5 \times 10^{-3} rad, 7m)$	**The first part**	56.39	13.69	99.49	13.55
	The second part	36.46	32.16	97.43	16.87
	The third part	41.96	10.90	19.91	21.08
$(8 \times 10^{-3} rad, 10m)$	**The first part**	86.85	20.97	128.23	35.27
	The second part	65.26	16.30	144.58	30.00
	The third part	91.61	16.05	35.74	41.96
$(11 \times 10^{-3} rad, 13m)$	**The first part**	135.35	87.48	193.12	68.19
	The second part	87.71	167.48	196.99	42.37
	The third part	107.32	99.23	57.10	74.23

can better analyze the effect of noise interference on different methods. Specifically, 4 kinds of observation noises with different deviation pairs $(\sigma_\theta, \sigma_d)$ are discussed: i.e., $(2\times10^{-3}rad, 4m)$, $(5\times10^{-3}rad, 7m)$, $(8\times10^{-3}rad, 10m)$ and $(11\times10^{-3}rad, 13m)$, respectively. All the tracking results with different observation noises are shown from Figure 19 to 22. In each figure, sub-figure (a) shows the position of trajectory tracking results, and sub-figures (b) and (c) show the RMSEs of position and velocity tracking, respectively. Specifically, the upper part of sub-figure (b) shows the RMSEs of position tracking in X-coordinate, the bottom part shows the ones in Y-coordinate. Similarly, the upper part of sub-figure (c) shows the RMSEs of velocity tracking in X-coordinate, the bottom part shows the ones in Y-coordinate. Considering that each part lasts 30s, we set the M=30 for RMSE calculation in (14.)

In those figures of low-maneuvering scene, the blue line with upper triangle denotes original trajectory, the black line with dot denotes the tracking results of DMTT, the dark green line with circle denotes the ones of CADP-NN, the pink line with plus denotes the ones of MIE-BLUE-IMM and the red line with star denotes the ones of ISPM. Moreover, the means of distance and velocity tracking RMSEs can be obtained in Tables IV and V. Generally speaking, the position tracking RMSEs[2] of ISPM are the smallest in comparison with other methods in the cases of observation noise deviations set to be $(2\times10^{-3}rad, 4m)$, $(5\times10^{-3}rad, 7m)$ and $(11\times10^{-3}rad, 13m)$, respectively. When the observation noise deviation is set to be $(8\times10^{-3}rad, 10m)$, DMTT performs better. For velocity tracking, DMTT performs better than ISPM except the observation noise is set to be $(2 \times 10^{-3}rad, 4m)$.

1) Analysis on the ability of motion-model estimation: The key for maneuvering tracking is to correctly and timely estimate the motion-model, especially when it changes. As we know in Table III, there are three motion-models in this scene. The first model changes to the second one at time 30s, and the second model changes to the third one at time 60s. As we can see from Figure 19 to 22, the RMSEs of CADP-NN algorithm always sharply become very large juts after 30s, which is the time when turn rate changes from $-3°/s$ to $8°/s$. Obviously, the CADP-NN algorithm cannot offer a good prediction on motion-model. From this point of view, DMTT plays better. But this problem still happens when the deviations of observation noise are $(2 \times 10^{-3}rad, 4m)$ and $(11 \times 10^{-3}rad, 13m)$, as we can see in Figures 19 and 22. There are obvious fluctuations in black line with dot after 30s. On the contrary, the fluctuations in pink line with plus and red line with star are small, especially at the time range 30-60s. That means, the idea of model-driven can gain more stability than data-driven. The basic idea of our ISPM is to estimate the motion-model with observation data, which contains both the accuracy of data-driven and stability of model-driven. Hence, our ISPM performs best in comparison with both data-driven and model-driven methods.

2) Analysis on noise interference: The anti-noise ability is another key factor in actual application, which guarantees the reliability and stability of the tracking algorithm. Under normal condition, the tracking errors increase along with the noises increase. Comparing ISPM with MIE-BLUE-IMM and CADP-NN, we can see that the position tracking RMSEs of ISPM increase from around 10m to 60m and the velocity tracking RMSEs increase from around 10m/s to 50m/s, along with the increment of noise deviation from $(2\times10^{-3}rad, 4m)$ to $(11 \times 10^{-3}rad, 13m)$. That is, the increment of RMSEs of position and velocity of ISPM tracking are 50m and 40m/s, respectively. But the increment of MIE-BLUE-IMM and CADP-NN are around 100m, 500m/s and 150m, 45m/s, respectively. Obviously, the RMSE increments of ISPM are in general smaller than MIE-BLUE-IMM and CADP-NN which verify that the anti-noise ability of our ISPM is stronger. The

[2]Position tracking RMSE is in fact similar to distance tracking RMSE.

TABLE V
MEANS OF VELOCITY TRACKING RMSES IN THREE SEGMENTS OF THE TRAJECTORY OF LOW-MANEUVERING SCENE.

Noises	Segments	MIE-BLUE-IMM (m/s)	DeepMTT (m/s)	CADP-NN (m/s)	ISPM (m/s)
$(2\times10^{-3}rad, 4m)$	**The first part**	50.87	12.32	28.35	8.91
	The second part	75.17	33.44	43.58	8.09
	The third part	20.59	8.24	6.53	12.90
$(5\times10^{-3}rad, 7m)$	**The first part**	291.57	12.48	47.80	25.72
	The second part	200.04	16.03	69.99	13.56
	The third part	214.77	7.23	13.41	25.55
$(8\times10^{-3}rad, 10m)$	**The first part**	483.86	10.78	60.35	35.04
	The second part	370.73	13.07	88.83	24.08
	The third part	453.92	8.15	26.78	46.90
$(11\times10^{-3}rad, 13m)$	**The first part**	768.66	19.05	71.54	52.34
	The second part	520.72	37.76	103.67	34.64
	The third part	599.18	24.93	34.83	70.62

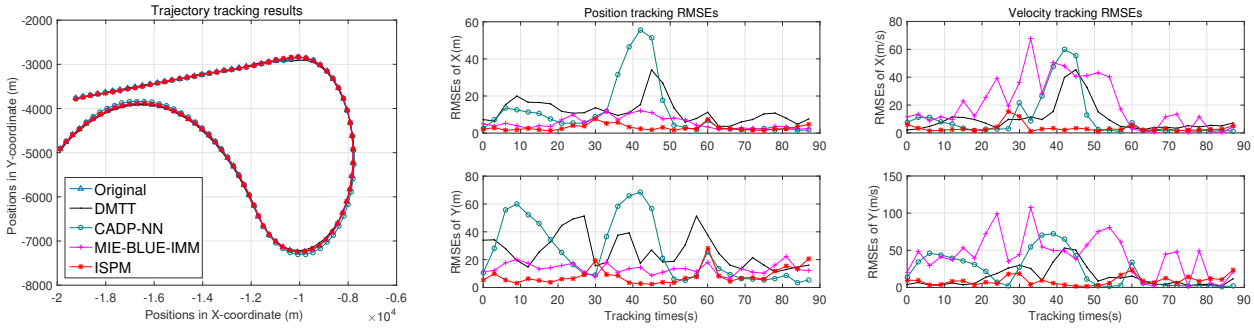
RMSEs of DMTT are special. The RMSEs of DMTT are smallest when the deviation of noise is $(8\times10^{-3}rad, 10m)$, as seen in Figure 21. When the deviation of noise increases to $(11\times10^{-3}rad, 13m)$, the RMSEs of DMTT seriously increase, especially the RMSEs of position tracking increase to round 150m after 30s, as seen in Figure 22. Moreover, the RMSEs of DMTT also increase when the deviation of noise decreases to $(2\times10^{-3}rad, 4m)$. That means, the effectiveness of DMTT is limited to a certain noise range. Considering that the DMTT is an end-to-end network to estimate the trajectory by observation information, those estimations of DMTT are in fact interfered by noise patterns. Hence, the DMTT does not have enough anti-noise ability.

C. Tracking for high-maneuvering scene

In this section, the tracking results of DMTT, CADP-NN, MIE-BLUE-IMM and ISPM algorithms in high-maneuvering scene are discussed. As we know in Figure 18, the trajectory in this scene is also in the range of ATC system, but the turn rates in segments 2, 3 and 5 is out of the range. To further analyze the interference of noise, we simulate the tracking on different deviations of observation noise: $(2\times10^{-3}rad, 4m)$, $(5\times10^{-3}rad, 7m)$ and $(8\times10^{-3}rad, 10m)$. The tracking results are shown from Figure 23 to 25. Comparing these figures with Figure 18, we can see that the DMTT, which is denoted by black line with dot, cannot correctly estimate the positions of trajectory from the second segment. Hence, it loses its tracking ability in such a high-maneuvering scene. The CADP-NN, MIE-BLUE-IMM and ISPM algorithms can provide effective tracking results even though the RMSEs of tracking become large. Moreover, considering that some parts only last 9s, we set the M=10 for RMSE calculation in (14) for this scene. To accurately compare the performances in different methods, the means of distance and velocity tracking RMSEs in each segment can be obtained in Tables VI and VII, respectively.

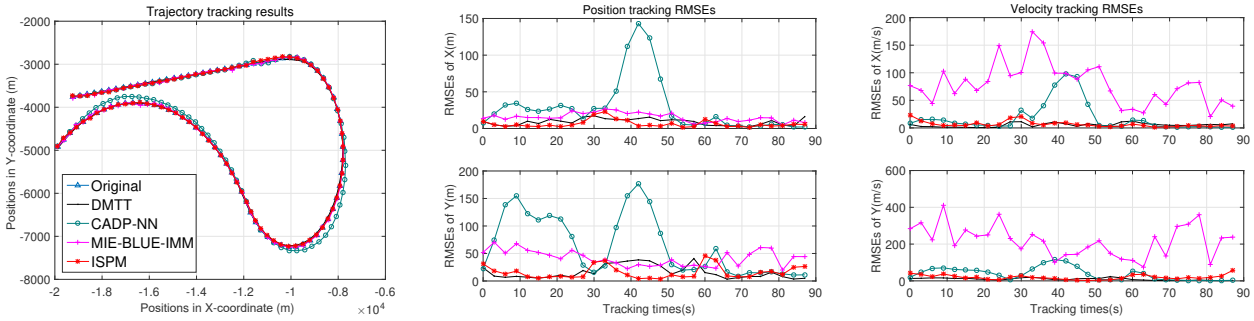
1) Analysis on the ability of motion-model estimation: In Table III, there are six motion-models in this scene. In this trajectory, the first model changes to the second one at time 21s, the second model changes to the third one at time 30s, the third model changes to the forth one at time 39s, the forth model changes to the fifth one at time 60s and the fifth model changes to the sixth one at time 69s, respectively. Obviously, as seen from Figure 23 to 25, at the times that models switch, the RMSEs of both CADP-NN and ISPM algorithms increase. But ISPM decreases more quickly and keeps a better tracking accuracy than CADP-NN. This can be obviously found in sub-figures (b) and (c) in Figures 23, 24 and 25 in the range of 60-70s. The RMSEs of our ISPM decreases from a high value to a small one by about 2s. Then the RMSEs keep small in the next around 6s. We know that this range of time belongs to segment 5 with turn rate $60°/s$. Hence our ISPM can correctly and timely identify this motion-model and make a good tracking. On the contrary, the RMSEs of CADP-NN keep high values. Hence, our ISPM outperforms CADP-NN on motion-model estimation in high-maneuvering scene. Notice that, MIE-BLUE-IMM is stablest for distance tracking benefiting from the characteristics of model-driven. But it fails in velocity tracking as seen in Table VII, i.e., it in fact has no ability to estimate the motion-model. Hence, in general, our ISPM has the best ability to estimate the motion-model.

2) Analysis on noise interference: Based on Figures 23 to 25 and Tables VI and VII, we can see that the RMSEs of all methods increase. Specifically, the RMSEs of model-driven method: MIE-BLUE-IMM increases slowly, but the ones of data-driven method: CADP-NN increases quickly. The proposed method: ISPM, which estimates the motion-model with observation data, has the moderate noise increment rate. For example, as seen in Table VI, when the deviation pairs of noise is $(2\times10^{-3}rad, 4m)$, the distance tracking RMSEs of CADP-NN in each segment are 31.99, 276.59, 274.44, 60.98, 279.94 and 108.70m, respectively. However, when the noise increases to $(8\times10^{-3}rad, 10m)$, the smallest RMSE has increased to 198.41m, and two RMSEs of segments exceed 500m. In contrast, in the scene of low noise, the distance tracking RMSEs of proposed ISPM are 13.78, 143.53, 180,82, 24.88, 141.82 and 48.87m, respectively. When the noise increases to $(8\times10^{-3}rad, 10m)$, there is only the RMSE of segment 3 exceeds 500m, and the others are smaller than 200m. Hence, the ISPM can better resist the increment of noise. Of course, MIE-BLUE-IMM performs better in distance tracking, but it has no ability to track the velocity information, as shown in Table VII. Hence, our ISPM has the best ability to resist the noise interference.



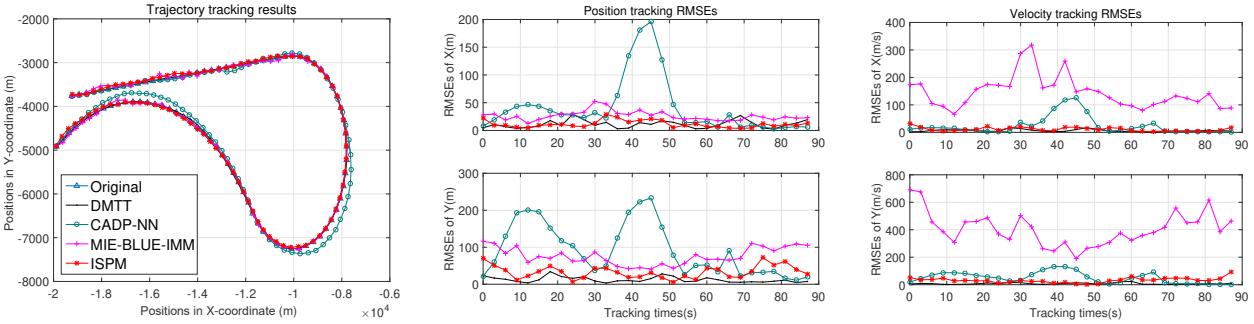
(a) (b) (c)

Fig. 19. Tracking results of low-maneuvering trajectory with $\sigma_\theta = 2 \times 10^{-3} rad$ and $\sigma_d = 4m$.



(a) (b) (c)

Fig. 20. Tracking results of low-maneuvering trajectory with $\sigma_\theta = 5 \times 10^{-3} rad$ and $\sigma_d = 7m$.



(a) (b) (c)

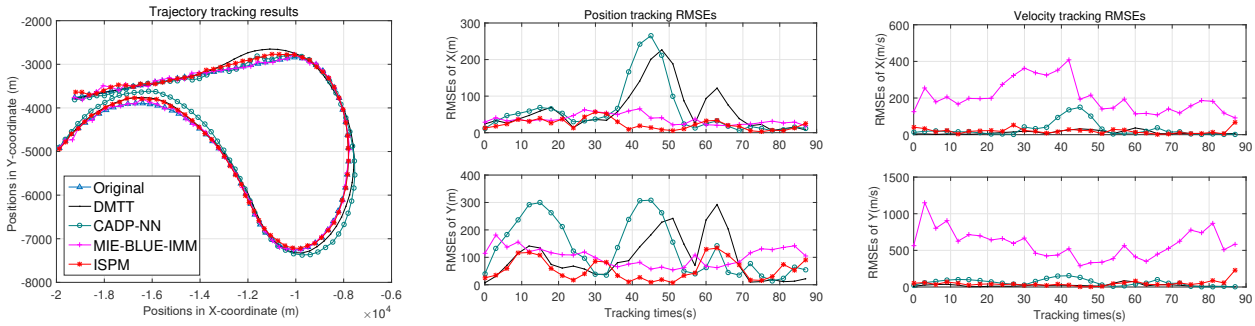
Fig. 21. Tracking results of low-maneuvering trajectory with $\sigma_\theta = 8 \times 10^{-3} rad$ and $\sigma_d = 10m$.

D. Ablation experiments

To further discuss the role of NEN in our ISPM, we test the tracking performance of ISPM without NEN in two low-maneuvering scenes, i.e., the ones with observation noises: $(5 \times 10^{-3} rad, 7m)$ and $(11 \times 10^{-3} rad, 13m)$, respectively. In comparison with ISPM, we can easily evaluate the values of NEN in the whole tracking process, as seen in Figure 26. The upper three subfigures describe the tracking performance on low-maneuvering scene with observation noise $(5 \times 10^{-3} rad, 7m)$. The bottom three subfigures describe the other one with observation noise $(11 \times 10^{-3} rad, 13m)$. Obviously, RMSEs of ISPM without NEN (black line with circle) are larger than the ones with NEN (red line with star). Moreover, we notice that the RMSEs in the bottom subfigures are much larger than the ones in upper subfigures. That means, when the observation noises increase, the RMSEs increase more seriously. But when we add the NEN for tracking, we can significantly slow down this growth trend. Hence, the NEN is necessary in our ISPM to effectively reduce noisy interference.

E. Computational complexity

In this subsection, the computational complexity of our ISPM method is discussed in comparison with DeepMTT, CADP-NN and MIE-BLUE-IMM methods, by means of test-



(a) (b) (c)

Fig. 22. Tracking results of low-maneuvering trajectory with $\sigma_\theta = 11 \times 10^{-3} rad$ and $\sigma_d = 13m$.

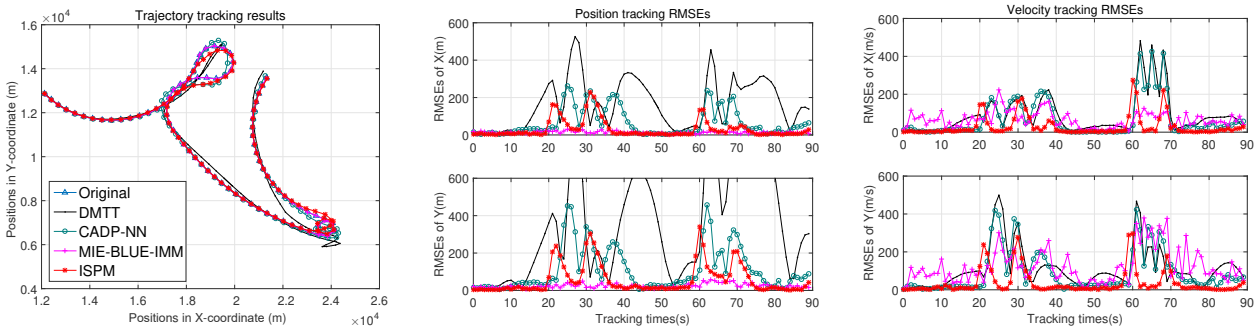
TABLE VI
MEANS OF DISTANCE TRACKING RMSES IN THREE SEGMENTS OF THE TRAJECTORY OF HIGH-MANEUVERING SCENE.

Noises	**Segments**	**MIE-BLUE-IMM** (m)	**DeepMTT** (m)	**CADP-NN** (m)	**ISPM** (m)
$(2 \times 10^{-3} rad, 4m)$	**The first part**	22.37	103.40	31.99	13.78
	The second part	35.94	614.00	276.59	143.53
	The third part	30.89	204.05	274.44	180.82
	The fourth part	21.20	371.62	60.98	24.88
	The fifth part	46.52	689.93	279.94	141.82
	The sixth part	27.47	626.64	108.70	48.87
$(5 \times 10^{-3} rad, 7m)$	**The first part**	58.33	20.58	102.01	26.01
	The second part	92.00	788.67	430.82	164.45
	The third part	93.27	391.75	297.35	386.86
	The fourth part	59.62	739.27	157.07	51.00
	The fifth part	101.02	1.43e+3	502.78	123.78
	The sixth part	79.10	4.95e+3	288.55	101.90
$(8 \times 10^{-3} rad, 10m)$	**The first part**	93.78	224.40	198.41	36.17
	The second part	156.84	808.03	501.67	156.44
	The third part	118.62	995.72	241.79	545.16
	The fourth part	132.43	3.26e+3	231.58	104.58
	The fifth part	134.49	1.47e+3	521.08	179.24
	The sixth part	137.08	6.53e+3	348.99	111.62

TABLE VII
MEANS OF VELOCITY TRACKING RMSES IN THREE SEGMENTS OF THE TRAJECTORY OF HIGH-MANEUVERING SCENE.

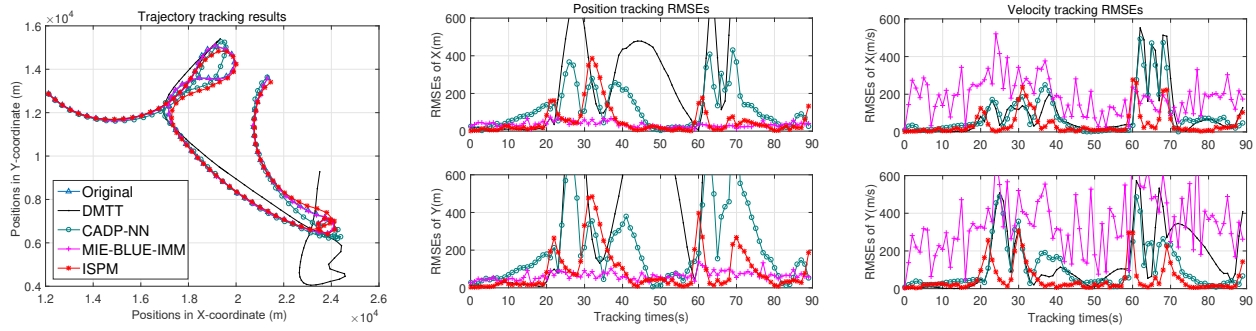
Noises	**Segments**	**MIE-BLUE-IMM** (m/s)	**DeepMTT** (m/s)	**CADP-NN** (m/s)	**ISPM** (m/s)
$(2 \times 10^{-3} rad, 4m)$	**The first part**	96.45	53.08	18.42	19.34
	The second part	206.59	308.09	286.09	102.22
	The third part	177.22	208.25	225.02	107.36
	The fourth part	78.75	78.50	33.36	29.84
	The fifth part	299.33	410.70	395.10	119.99
	The sixth part	139.80	115.85	75.24	30.70
$(5 \times 10^{-3} rad, 7m)$	**The first part**	299.87	14.55	41.89	29.62
	The second part	532.82	302.82	300.63	118.53
	The third part	527.14	163.77	234.44	159.75
	The fourth part	301.69	84.01	66.68	39.39
	The fifth part	615.45	518.77	431.30	138.09
	The sixth part	401.10	255.52	105.78	58.11
$(8 \times 10^{-3} rad, 10m)$	**The first part**	536.77	62.91	66.47	35.59
	The second part	1.02e+3	339.92	302.36	130.42
	The third part	682.61	182.16	221.39	196.76
	The fourth part	677.79	151.73	72.26	70.39
	The fifth part	831.47	589.19	430.26	180.02
	The sixth part	805.40	511.30	152.04	56.87

ing the computational time in single iteration of tracking. For fair comparison, we test all the methods with the same Intel Core i7-3770 CPU at 3.4 GHz and 4 GB RAM. In the tracking process, our ISPM method consumes 17.8 ms to compute one iteration in tracking process. According to [15], [19], DeepMTT, CADP-NN and MIE-BLUE-IMM methods consume 16.4 ms, 2.0ms and 0.6 ms, respectively. Although the computational time of our ISPM for one iteration of tracking is the longest in comparison with other methods, 17.8ms per iteration is in fact short enough for practical



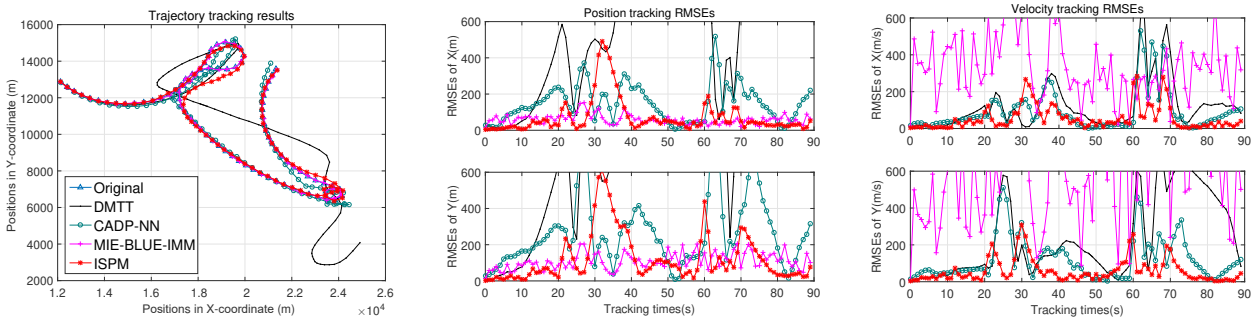
(a) (b) (c)

Fig. 23. Tracking results of high-maneuvering trajectory with $\sigma_\theta = 2 \times 10^{-3} rad$ and $\sigma_d = 4m$.



(a) (b) (c)

Fig. 24. Tracking results of high-maneuvering trajectory with $\sigma_\theta = 5 \times 10^{-3} rad$ and $\sigma_d = 7m$.



(a) (b) (c)

Fig. 25. Tracking results of high-maneuvering trajectory with $\sigma_\theta = 8 \times 10^{-3} rad$ and $\sigma_d = 10m$.

applications of maneuvering-target tracking.

VI. Conclusion

In this paper, a digital twin system has been built to explore the data features of noisy observations and their relationship to target states. Based on those features, two neural networks have been constructed. One is the noise eliminated network which can effectively eliminate the noise and rebuild the noise-limited trajectory by a transformer + LSTM structure. The other is the motion-model estimation network which uses the CADP information and CNN structure to estimate the transition matrix by observations. Then those two networks are combined into our intelligent state prediction method (ISPM) to track the maneuvering target. Simulation results have conducted to verify the validity of our theoretical analysis and network structure, in which one can see that the proposed ISPM outperforms the state-of-the-art maneuvering-target tracking methods in both low-maneuvering and high-maneuvering scenes.

Although the tracking accuracy and generalization ability have been greatly improved, there is still a lot of room for improvement in tracking performance, especially in high-maneuvering scenes. Another future work is that the three-dimensional space for tracking needs to be considered, by

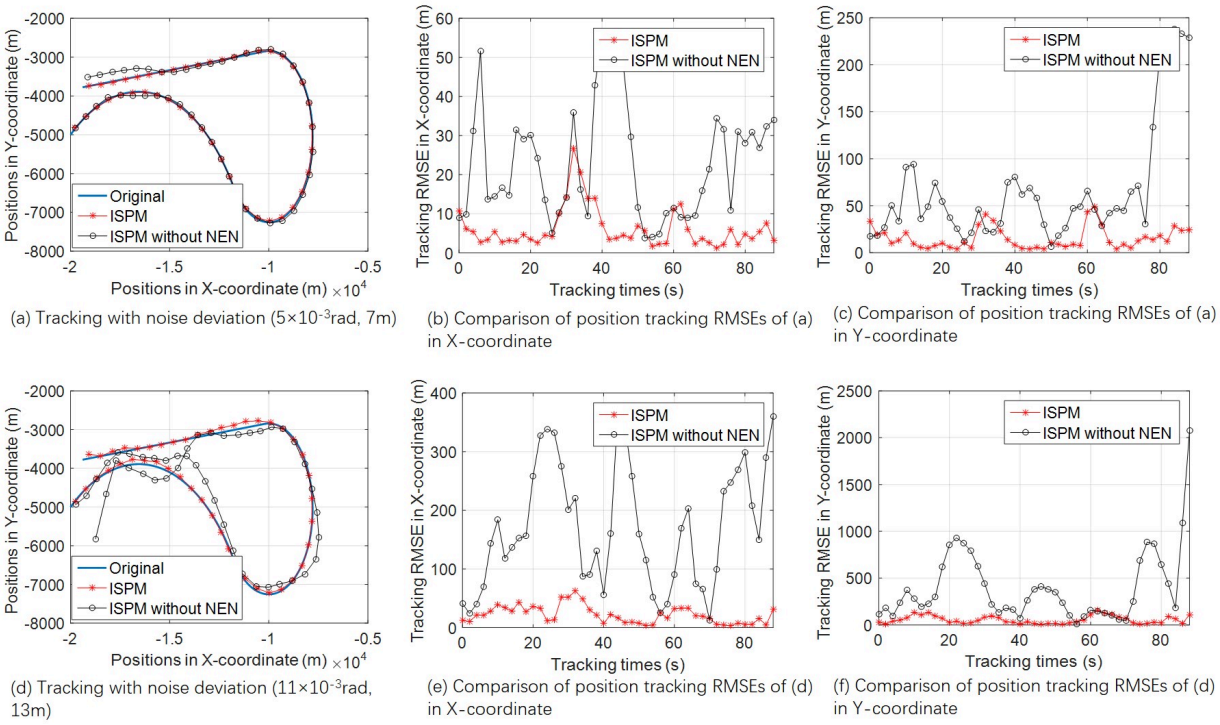


Fig. 26. Ablation experiments.

which those intelligent maneuvering-target tracking methods can be more suitable for practical application.

REFERENCES

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

[2] S. J. Julier and J. K. Uhlmann, "A new extension of the kalman filter to nonlinear systems," *Proc.aerosense Int.symp.aerospace defence Sensing Simulation and Controls*, 1997.

[3] S. Julier, J. Uhlmann, and H. F. Durrantwhyte, "A new method for the nonlinear transformation of means and covariances in filters and estimators," *IEEE Transactions on Automatic Control*, vol. 45, no. 3, pp. 477–482, 2000.

[4] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking," *IEEE Trans. Signal Process.*, vol. 50, no. 2, pp. 174–188, 2002.

[5] J. Liu, Z. Wang, and M. Xu, "A novel distributed fusion algorithm for multi-sensor nonlinear tracking," *EURASIP J. Adv. Signal Process*, vol. 2016, no. 1, pp. 1–11, 2016. [Online]. Available: http://dx.doi.org/10.1186/s13634-016-0362-y

[6] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017.

[7] S. S. Kim, S. I. Chien, C. K. An, S. D. Lee, and T. H. Lee, "Recurrent neural network," *KITE JOURNAL OF ELECTRONICS ENGINEERING*, vol. 5, no. 1, 1994.

[8] 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.

[9] 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.

[10] V. Petridis and A. Kehagias, "A multi-model algorithm for parameter estimation of time-varying nonlinear systems," *Automatica*, vol. 34, no. 4, pp. 469–475, 1998.

[11] X. R. Li, Z. Zhao, and X. B. Li, "General model-set design methods for multiple-model approach," *IEEE Transactions on Automatic Control*, vol. 50, no. 9, pp. 1260–1276, 2005.

[12] X. R. Li, "Multiple-model estimation with variable structure. ii. model-set adaptation," *IEEE Transactions on Automatic Control*, vol. 45, no. 11, pp. 2047–2060, 2000.

[13] L. Xu, X. R. Li, and Z. 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.

[14] H. Sheng, W. Zhao, and J. Wang, "Interacting multiple model tracking algorithm fusing input estimation and best linear unbiased estimation filter," *Iet Radar Sonar and Navigation*, vol. 11, no. 1, pp. 70–77, 2017.

[15] J. Liu, Z. 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.

[16] C. Gao, J. Yan, S. Zhou, P. K. Varshney, and H. Liu, "Long short-term memory-based deep recurrent neural networks for target tracking," *Information Sciences*, vol. 502, pp. 279–296, 2019.

[17] H. Liu, L. Xia, and C. Wang, "Maneuvering target tracking using simultaneous optimization and feedback learning algorithm based on elman neural network," *Sensors (Basel, Switzerland)*, vol. 19, no. 7, 2019.

[18] L. Song, W. Shengli, and X. Dingbao, "Radar track prediction method based on bp neural network," *The Journal of Engineering*, vol. 2019, no. 21, pp. 8051–8055, 2019.

[19] J. Liu, S. Yang, and F. Yang, "A cross-and-dot-product neural network based filtering for maneuvering-target tracking," *Neural Computing and Applications*, 2022.

[20] X. R. Li and V. P. Jilkov, "Survey of maneuvering target tracking. part i. dynamic models," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 39, no. 4, pp. 1333 – 1364, 2003.

[21] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional lstm and other neural network architectures," *Neural Networks*, vol. 18, no. 5-6, pp. 602–610, 2005.

[22] A. Graves *et al.*, *Supervised sequence labelling with recurrent neural networks*. Springer, 2012, vol. 385.

[23] Y. Lecun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, 1989.

Jingxian Liu received the Ph.D. degree from Beihang University, Beijing, China, in 2018. He is currently an associate research fellow with Guangxi University of Science and Technology, Guangxi, China. His research interests include signal processing with artificial intelligence, radar target detection and tracking, and computer vision.

Junjie Yan received the M.S. and Ph.D. degrees from the Chongqing University of Posts and Telecommunications, Chongqing, China, in 2016 and 2020, respectively. He is a lecturer with the Guangxi University of Science and Technology at Liuzhou, China. His research interests include D2D communication, MEC, UAV communication.

Dehuan Wan received the Ph.D. degree from South China University of Technology, Guangzhou, China, in 2018. He is currently an Associate Professor with Guangdong University of Finance, Guangzhou, China. His current research interests include wireless communications and data science and artificial intelligence.

Xuran Li received the M.Sc. degree in communication engineering and the Ph.D. degree in electronic information technology from the Faculty of Information and Technology of Macau University of Science and Technology (MUST) in 2015 and 2018, respectively. He is currently a lecturer in School of Physics and Electronics, Shandong Normal University (SDNU), China. His research interests include wireless communication networks, security of wireless networks, and the Internet of Things.

Saba Al-Rubaye (Senior Member, IEEE) , Professor Chair in Telecommunications and Autonomous Systems in the School of Aerospace, Transport, and Manufacturing at Cranfield University. With over 20 years of experience in both industry and academia, Prof Al-Rubaye has a proven track record in initiating innovative solutions across various domains, including design, testing, consultation, leadership, and program development. She has actively participated in numerous projects sponsored by prestigious organizations such as Innovation UK, Research England, EPSRC, the Department for Transport, the Natural Sciences and Engineering Research Council of Canada (NSERC), and the USA Government. An essential contribution made by Prof Al-Rubaye was her involvement in establishing the hardware in the loop testbed for smart grid communications and system integration at Quanta's Sustainable Technology Integration Laboratory (QT-STIL) in Toronto, Canada. These projects are significant steps towards expediting the adoption of emerging technology solutions and their integration into future communications networks. Prof Al-Rubaye actively participates in the development of industry standards, contributing as a voting member in IEEE P1932.1 standard for License/Unlicensed Interoperability and IEEE P1920.2 standard for Vehicle-to-Vehicle Communications for Unmanned Aircraft Systems. Her research work has been published in various prestigious IEEE journals and conferences, earning her the distinguished honour of receiving the best technical paper award in 2011 and 2015, respectively, in IEEE Vehicular Technology as well as best paper in IEEE DASC conference 2020 and 2021. Furthermore, Prof Al-Rubaye has taken on leadership roles in several international conferences, including serving as a general co-chair and TPC co-chair. She has also delivered tutorials/Talk at esteemed IEEE conferences such as ICC and VTC, as well as delivering invited talks at reputable venues like the Communications Research Centre (CRC) of Canada, IEEE Toronto Chapter, and IEEE UK Chapter. Prof Saba Al-Rubaye is a distinguished professional Chartered Engineer (CEng), MIET, and a Life

Anwer Al-Dulaimi (Senior Member, IEEE) is currently a Senior Manager of Emerging technologies and Distinguished Member of Technical Staff in the 5G Center of Excellence, EXFO, Montreal, Canada. He received the Ph.D. degree in electrical and computer engineering from Brunel University, London, U.K., in 2012 after obtaining M.Sc. and B.Sc. honors degrees in telecommunication engineering. He was a Postdoctoral Fellow in the Department of Electrical and Computer Engineering, University of Toronto, sponsored by Blackberry's advanced research team. In his current role, Anwer is responsible for identifying new trends of mobile technology evolutions and the adaptation phases that EXFO need to take for compliancy. He is the chair of the newly established IEEE 5G/6G Innovation Testbed Project working to develop a convergent platform for E2E network innovations. He is the chair of the IEEE 1932.1 'Standard for Licensed/Unlicensed Spectrum Interoperability in Wireless Mobile Network'. He is also representing EXFO in many industrial forums such as One6G and other collaborative projects. He has published many papers, edited books, and developed patents focusing on new generations of mobile networking technologies. He is the Editor of IEEE Future Networks Series on 5G & 6G published by IEEE Vehicular Technology Magazine, editor of Vehicular Networking Series in IEEE Communication Standards Magazine and guest editor of many other IEEE series issues. His research interests include 5G/6G networks, cloud computing, IoT and cybersecurity. He is a Fellow of the Institution of Engineering and Technology (FIET), registered as a Chartered Engineer (CEng) by the British Engineering Council since 2010. He is a senior member of IEEE and ComSoc Distinguished Lecturer as well as voting member of Mobile Communication Networks Standards Committee (MobiNet-SC).

Zhi Quan (Senior Member, IEEE) is a distinguished professor with the College of Electronics and Information Engineering, Shenzhen University, China. He received his Ph.D. in Electrical Engineering from University of California, Los Angeles (UCLA) with highest honors in 2009, and his B.E. in Communications Engineering from Beijing University of Posts and Telecommunications (BUPT), China in 1999. He worked as a Sr. System Engineer in Qualcomm Research Center (QRC) of Qualcomm Inc. (San Diego, CA) during 2008-2012, and as a RF System Architect with Apple Inc. (Cupertino, CA) during 2012-2015. Dr. Quan had contributed to IEEE 802.11ac/ah standards with over 30 U.S. issued patents and published over 60 papers in wireless communications and signal processing with more than 5000 citations from Google Scholar. Dr. Quan was the recipient of UCLA Outstanding Ph.D. Award in 2009, IEEE Signal Processing Society Best Paper Award in 2012, China National Excellent Young Scientist Foundation in 2016, and First Prize Technology Innovation Award by China Institute of Communications in 2020. His current research interests include wireless communication systems, RF system calibration and measurement, data-driven signal processing, and machine learning.

2023-08-30

Digital twins based intelligent state prediction method for maneuvering-target tracking

Liu, Jingxian

IEEE

Liu J, Yan J, Wan D, et al., (2023) Digital twins based intelligent state prediction method for maneuvering-target tracking. IEEE Journal on Selected Areas in Communications, Available online 30 August 2023

https://doi.org/10.1109/JSAC.2023.3310109