Article

Deep-Learning-Based Multiple Model Tracking Method for Targets with Complex Maneuvering Motion

Weiming Tian 1, Linlin Fang 1, Weidong Li 1,2,* , Na Ni 1 , Rui Wang 1,2 , Cheng Hu 1,2 , Hanzhe Liu 1 and Weigang Luo 1

1 Radar Research Lab, School of Information and Electronics, Beijing Institute of Technology, Beijing 100081, China; tianweiming@bit.edu.cn (W.T.); 3120185412@bit.edu.cn (L.F.); 3120205379@bit.edu.cn (N.N.); wangui.bit@bit.edu.cn (R.W.); cchchb@bit.edu.cn (C.H.); 3220200584@bit.edu.cn (H.L.); 3220200601@bit.edu.cn (W.L.)
2 Advanced Technology Research Institute, Beijing Institute of Technology, Jinan 250300, China
* Correspondence: lwd.bit@bit.edu.cn

Abstract: The effective detection of unmanned aerial vehicle (UAV) targets is of great significance to guarantee national military security and social stability. In recent years, with the development of communication and control technology, the movement of UAVs has become increasingly flexible and complex, presenting diverse trajectory forms and different motion models in different phases. The Gaussian mixture probability hypothesis density filter incorporating the linear Gaussian jump Markov system approach (LGJMS-GMPhD) provides an efficient method for tracking multiple maneuvering targets, as applied to the switching of motions between a set of models in a Markovian chain. However, in practice, the motion model parameters of targets are generally unknown and the model switching is uncertain. When the preset filtering model parameters are mismatched, the tracking performance is dramatically degraded. In this paper, within the framework of the LGJMS-GMPhD filter, a deep-learning-based multiple model tracking method is proposed. First, an adaptive turn rate estimation network is designed to solve the filtering model mismatch caused by unknown turn rate parameters in coordinate turn models. Second, a filter state modification network is designed to solve the large tracking errors in the maneuvering phase caused by uncertain motion model switching. Finally, based on simulations of multiple maneuvering targets in cluttered environments and experimental field data verification, it can be concluded that the proposed method has strong adaptability to multiple maneuvering forms and can effectively improve the tracking performance of targets with complex maneuvering motion.

Keywords: deep learning; multiple model; maneuvering target tracking

1. Introduction

Unmanned aerial vehicles (UAVs), with their advantages of flexible usage and easy deployment, have been widely used in many fields, including military applications such as battlefield surveillance, air reconnaissance, and tracking, as well as in civilian applications such as disaster monitoring, geophysical exploration, and agricultural plant protection [1]. At the same time, their malicious use also poses a severe threat to aviation security and the maintenance of social stability [2]. Radar has proven to be an essential technical method of UAV detection [3,4]. However, the strong maneuverability of UAVs, including their unknown motion model parameters and uncertain model switching [5], brings about great challenges in the stable tracking of targets.

Random finite set (RFS)-based filters provide a theoretically optimal approach to multi-target tracking and are widely used in many fields [6–8]. In the RFS formulation, the collection of target states at any given time is regarded as a set-valued multi-target state, and the collection of measurements is regarded as a set-valued multi-target observation [9,10]. Then, the problem of dynamically estimating multiple targets can be
formulated in a Bayesian filtering framework by propagating the posterior distribution of the multi-target state in time. However, multi-target filtering solutions in their standard form are generally implemented by a fixed motion model. Thus, to characterize the a rapidly maneuvering target with multiple models, the jump Markov system (JMS) approach has been incorporated into the existing RFS-based filters. Mahler derived the probability hypothesis density (PHD) filter and cardinalized probability hypothesis density (CPHD) filter approximations of the jump Markov multitarget Bayes filter [11]. On this basis, a closed-form solution to the PHD recursion for the linear Gaussian jump Markov multitarget model was derived, and this is referred as the LGJMS-GMPHD filter [12]. In addition, the JMS approach has also been incorporated into CPHD [13], multi-Bernoulli [14], and labeled multi-Bernoulli filters [7]. However, in the above multiple model filters, the constant velocity (CV) and coordinate turn (CT) models are generally combined as model sets to describe the maneuvering motion. When these are applied to practical UAV tracking scenarios, there are two key problems: (i) the turn rate of the target is generally unknown, resulting in a mismatch between the target’s motion and the CT filtering model; (ii) for complex maneuvering trajectories, the uncertainty of motion model switching leads to the degradation of tracking performance.

There are two common approaches to estimating the turn rate. The first class estimates the turn rate online by using the estimated acceleration magnitude over the estimated speed [15]. However, the estimated acceleration is generally not accurate enough, leading to inaccurate turn rate estimates. For the second class, the turn rate parameter is augmented into the state vector and the turn rate is estimated as part of the state vector recursively, which creates a difficult nonlinear problem [16]. Furthermore, target Doppler measurements usually have higher precision compared with position estimates, and thus they have been incorporated to improve the accuracy of the turn rate estimate. In [17], four possible turn rates were estimated based on the Doppler measurement. Due to the lack of prior information on target motion, the minimum turn rate and its opposite value were chosen to be the possible turn rates, and an interacting multiple model (IMM) algorithm consisting of one CV model and two CT models was adopted, increasing the computational burden.

The uncertainty of motion model switching is also a challenging problem in multiple model (MM) filters. In this case, the filtering state estimate has a delay in its response to target maneuvers, so the tracking precision degrades rapidly in the maneuvering phase. A smoothing process can produce delayed estimates, i.e., obtaining the target state estimates at time $k$ given the measurements up to time $k+d (d > 0)$, which contributes to improving the tracking performance of maneuvering targets and has been incorporated into the MM algorithms [18–20]. A sequential Monte Carlo (SMC) implementation for MMPHD smoothing was derived in [21] to improve the capability of PHD-based tracking algorithms. However, the backward smoothing procedure is still model-based and cannot extract temporal correlations, so longer lags do not yield better estimates of the current state. Moreover, closed-form recursion with a smoothing process needs to be derived for different filters respectively and the computation is generally intractable, leading to limitations in its applications. Deep neural networks have strong capability of fitting if there are sufficient training data [22], which is conducive to solving the problems of model mismatch and estimation delays in existing MM filtering algorithms for targets with complex maneuvering motions. In [23], a deep learning maneuvering target tracking algorithm was proposed. However, the input of the network was the filtering states estimated by a single-model unscented Kalman filter. For strong maneuvering trajectories, large tracking errors increase the training burden of the network and reduce the convergence rate of training losses. Additionally, an LSTM-based deep recurrent neural network was presented in [24] to overcome the limits of traditional model-based methods, but only CV and constant acceleration (CA) motions were verified, without considering CT
maneuvers. Furthermore, the above methods are not suitable for scenarios with multiple maneuvering targets.

In view of the above problems, within the framework of the LGJMS-GMPHD filter, we propose a deep-learning-based multiple model tracking method for multiple maneuvering targets with complex maneuvering motions. The main contributions of this study are summarized as follows:

- An adaptive turn rate estimation network (ATN) is designed. The feature matrix, including multi-frame and multi-dimensional kinematic states, is constructed and used as the input of the network. The temporal correlation information at each previous time step and the weight of different variables are extracted to improve the estimation accuracy of the turn rate. Then, the parameter is fed back to the CT model to enhance the consistency of the target motion and filtering model.

- A filter state modification network (FMN) is designed to smooth the state estimates of the filter. The relationship information between all time steps of the state vector, including position estimates, is extracted to improve the adaptability of the filter to complex maneuvering motions, thereby reducing the tracking errors caused by uncertain model switching.

The remainder of this paper is structured as follows. Section 2 reviews the PHD recursion for the linear Gaussian jump Markov multi-target model. Section 3 presents the principle of the proposed deep-learning-based multiple model tracking method. Section 4 designs the network parameters and verifies the effectiveness of the method in improving tracking performance. The simulation results and experimental data verification are analyzed in Section 5. The discussion and future work are presented in Section 6, and our conclusions are presented in Section 7.

2. Principle of LGJMS-GMPHD

The PHD filter [25] recursively propagates the first order moment or the intensity function associated with the multi-target posterior density, which has the advantage of low computational complexity and has been widely used. Furthermore, a closed-form solution to the PHD recursion for LGJMS derived in [12] provides an efficient method for tracking multiple maneuvering targets. In this section, a short review of RFS and the PHD recursion for the linear Gaussian jump Markov multi-target model is presented.

2.1. Random Finite Sets in Multi-Target Tracking

(1) Dynamic model

For a given multi-target state $X_{k-1}$ at time $k-1$, each $x_{k-1} \in X_{k-1}$ either continues to exist at time $k$ with probability $p_{S,k}(x_{k-1})$, or dies with probability $1 - p_{S,k}(x_{k-1})$. Consequently, at the next time step, the behavior of the given state $x_{k-1} \in X_{k-1}$ is modelled as the RFS

$$S_{dit-1}(x_{k-1})$$

(1)

which can take on either $\{x_i\}$ when the target survives, or $\emptyset$ when the target dies. For the sake of simplicity, spawned targets are not considered here. The multi-target state $X_k$ at time $k$ is given by the union of the surviving targets and the spontaneous births:

$$X_k = \bigcup_{x_{k-1} \in X_{k-1}} S_{dit-1}(x_{k-1}) \bigcup \Gamma_k,$$

(2)

where $\Gamma_k$ is the RFS of spontaneous birth at time $k$. Each target state $x_i = \begin{bmatrix} x_i & \dot{x}_i & y_i & \dot{y}_i \end{bmatrix}^T$ denotes the position and velocity of the $x$-axis and $y$-axis.
For the linear Gaussian multi-target model [12], the multi-target transition density is represented as:

\[ f_{x_{k-1}}(x_k | x_{k-1}) = \mathcal{N}(x_k; F_{x_{k-1}}x_{k-1}, Q_{x_{k-1}}), \]

where \( \mathcal{N} \) denotes a Gaussian density with mean \( m \) and covariance \( P \); \( F_{x_{k-1}} \) is the state transition matrix; \( Q_{x_{k-1}} \) is the process noise covariance.

In general, motion along a fixed heading at constant speed can be described by a CV model, and a level turn can be described by a CT model. At a turn rate of \( 10 \text{s}^{-1} \), the CT model reduces to the CV model. Therefore, the transition matrix can be uniformly expressed as:

\[
F_{x_{k-1}} = \begin{bmatrix}
1 & \sin \omega T & 0 & -\frac{1 - \cos \omega T}{\omega} \\
0 & \cos \omega T & 0 & -\sin \omega T \\
0 & 1 - \cos \omega T & 1 & \sin \omega T \\
0 & \sin \omega T & 0 & \cos \omega T
\end{bmatrix},
\]

where \( T \) is the sample interval and \( \omega \) is the turn rate, which is defined as the rate of change of the (velocity) heading angle in the horizontal plane and estimated as the magnitude of the acceleration divided by the speed of the target [15]. In this paper, we refer to this method as the physical definition method (PDM).

(2) Measurement model

The RFS measurement model, which accounts for detection uncertainty and clutter, is described as follows. A given target \( x_k \in X_k \) is either detected with probability \( p_{D,k}(x_k) \) or missed with probability \( 1 - p_{D,k}(x_k) \). At time \( k \), each state \( x_k \in X_k \) generates an RFS

\[ \Theta_k(x_k) \]

that can take on either \( \{ z_k \} \) when the target is detected, or \( \emptyset \) when the target is not detected. Given a multi-target state \( X_k \) at time \( k \), the multi-target measurement \( Z_k \) received at the sensor is formed by the union of target generated measurements and clutter, i.e.,

\[ Z_k = K_k \bigcup_{x \in X_k} \Theta_k(x) \]

For the linear Gaussian multi-target model, the multitarget likelihood is represented as:

\[ g_k(z_k | x_k) = \mathcal{N}(z_k; H_kx_k, R_k), \]

where \( H_k \) is the observation matrix and \( R_k \) is the measurement noise covariance.

Target Doppler measurements may provide additional information about the target’s kinematic state and therefore can be incorporated into the tracker to enhance the tracking performance [26]. After the Doppler is introduced, each measurement at time \( k \) is denoted as \( z_k = \begin{bmatrix} x_k^w & y_k^w & \vec{r}_k^w \end{bmatrix} \), where \( x_k^w \) and \( y_k^w \) are the positions of \( x \)-axis and \( y \)-axis, and \( \vec{r}_k^w \) is the Doppler. The modified measurement equation is expressed as follows:
\[
\begin{align*}
\mathbf{z}_k &= \mathbf{h}_k (\mathbf{x}_k) + \mathbf{n}_k = \begin{bmatrix} x_k^m \\ y_k^m \\ \mathbf{r}_k^m \end{bmatrix} = \begin{bmatrix} x_k \\ y_k \\ n_k^x \\ n_k^y \end{bmatrix}, \\
\mathbf{r}_k &= \mathbf{h}_{d,k} (\mathbf{x}_k) = \frac{x_k \mathbf{x}_k + y_k \mathbf{y}_k}{\sqrt{x_k^2 + y_k^2}},
\end{align*}
\]

where \( \mathbf{n}_k \) is Gaussian measurement noise with mean zero and covariance \( \mathbf{R}_k = \text{diag} \left( \sigma_x^2, \sigma_y^2, \sigma_d^2 \right) \). \( \sigma_x^2 \) and \( \sigma_y^2 \) are the position noise variance, and \( \sigma_d^2 \) is the Doppler noise variance. Obviously, the Doppler measurement function \( \mathbf{h}_{d,k} (\cdot) \) is nonlinear.

In this paper, to simplify the operation, we take the Taylor series expansion of (9) at \( \mathbf{x}_k \) to linearize the measurement equation. Then, the observation matrix is modified as:

\[
\begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
\frac{\partial \mathbf{r}_k}{\partial x_k} & \frac{\partial \mathbf{r}_k}{\partial y_k} & \frac{\partial \mathbf{r}_k}{\partial \mathbf{x}_k} & \frac{\partial \mathbf{r}_k}{\partial \mathbf{y}_k}
\end{bmatrix}.
\]

where

\[
\begin{align*}
\frac{\partial \mathbf{r}_k}{\partial x_k} &= \left( x_k^2 y_k^2 - x_k y_k \mathbf{x}_k \right) \left( x_k^2 + y_k^2 \right)^{3/2}, \\
\frac{\partial \mathbf{r}_k}{\partial y_k} &= \left( y_k^2 x_k^2 - x_k y_k \mathbf{y}_k \right) \left( x_k^2 + y_k^2 \right)^{3/2}, \\
\frac{\partial \mathbf{r}_k}{\partial \mathbf{x}_k} &= x_k \left( x_k^2 + y_k^2 \right)^{1/2}, \\
\frac{\partial \mathbf{r}_k}{\partial \mathbf{y}_k} &= y_k \left( x_k^2 + y_k^2 \right)^{1/2}.
\end{align*}
\]

2.2. PHD Recursion for LGJMS Multi-Target Model

JMS provides a natural means to model a maneuvering target for which the behavior cannot always be characterized by a single model. It is described by a set of parameterized state space models, the underlying parameters of which evolve with time according to a finite state Markov chain [12]. Suppose that \( \mu_k \in \mathcal{M} \) is the label of the model in effect at time \( k \), where \( \mathcal{M} \) denotes the (discrete) set of all model labels. The models follow a discrete Markov chain with transition probability \( t_{\mu_k,\mu_{k-1}} \). The state vector is augmented as \( \mathbf{x}_k = \left[ \xi_k^T, \mu_k \right]^T \in \mathbb{R}^n \times \mathcal{M} \) and the transition is governed by

\[
f_{\mu_{k-1}} \left( \mathbf{x}_k | \mathbf{x}_{k-1} \right) = \tilde{f}_{\mu_{k-1}} \left( \xi_k | \xi_{k-1}, \mu_k \right) t_{\mu_{k-1}} \left( \mu_k | \mu_{k-1} \right).
\]

The corresponding measurement likelihood is

\[
g_{\mu_k} \left( \xi_k | \mathbf{x}_k \right) = g_{\mu_k} \left( \xi_k | \xi_{k-1}, \mu_k \right).
\]

For a JMS with linear Gaussian models, the state transition density and measurement likelihood conditioned on mode \( \mu_k \) are given by

\[
\tilde{f}_{\mu_{k-1}} \left( \xi_k | \xi_{k-1}, \mu_k \right) = \mathcal{N} \left( \xi_k | \mathbf{F}_{k-1} \left( \mu_k \right) \xi_{k-1}, \mathbf{Q}_{k-1} \left( \mu_k \right) \right),
\]

\[\text{(14)}\]
\[ g_{k}(z_{k} \mid \xi_{k}, \mu_{k}) = N(z_{k} \mid H_{k} (\mu_{k}) \xi_{k}, R_{k} (\mu_{k})) , \]  

(15)

where \( F_{k-1}(\mu_{k}) \) and \( H_{k}(\mu_{k}) \) denote the transition and the observation matrices of model \( \mu_{k} \), and \( Q_{k-1}(\mu_{k}) \) and \( R_{k}(\mu_{k}) \) denote covariance matrices of the process noise and measurement noise.

The purpose of maneuvering target tracking is to estimate the augmented state \( x_{i} \) at time \( k \) from the sequence of measurement sets \( z_{i,k} = (z_{i}, \ldots, z_{i}) \). The closed-form PHD recursion for the LGJMS multi-target model [12] is presented as follows:

- **Prediction step**

  For an LGJMS multi-target model, if the posterior intensity \( v_{k-1} \) at time \( k-1 \) has the form

  \[ v_{k-1}(\xi_{k-1}, \mu_{k-1}) = \sum_{\mu_{j-1}} \sum_{j=1}^{h_{k-1}(\mu_{j-1})} w_{k-1}^{(j)}(\mu_{k-1}) N(\xi_{k-1}; m_{k-1}^{(j)}(\mu_{k-1}), P_{k-1}^{(j)}(\mu_{k-1})) , \]  

  (16)

  then the predicted intensity \( v_{k,k-1} \) is given by

  \[ v_{k,k-1}(\xi_{k}, \mu_{k}) = v_{S,k,k-1}(\xi_{k}, \mu_{k}) + v_{B,k}(\xi_{k}, \mu_{k}) , \]  

  (17)

  where \( v_{B,k}(\xi_{k}, \mu_{k}) \) is the intensity of birth targets, and the intensity of survival targets is represented as:

  \[ v_{S,k,k-1}(\xi_{k}, \mu_{k}) = \sum_{\mu_{j-1}} \sum_{j=1}^{h_{k,k-1}(\mu_{j-1})} w_{S,k,k-1}^{(j)}(\mu_{k}, \mu_{k-1}) N(\xi_{k}; m_{k,k-1}^{(j)}(\mu_{k}, \mu_{k-1}), P_{k,k-1}^{(j)}(\mu_{k}, \mu_{k-1})). \]  

  (18)

  The weight, mean, and covariance of the survival Gaussian density are represented as:

  \[ w_{S,k,k-1}^{(j)}(\mu_{k}, \mu_{k-1}) = \tau_{k,k-1}(\mu_{k} \mid \mu_{k-1}) p_{B,k}(\mu_{k-1}) w_{k-1}^{(j)}(\mu_{k-1}) , \]  

  (19)

  \[ m_{S,k,k-1}^{(j)}(\mu_{k}, \mu_{k-1}) = F_{S,k,k-1}(\mu_{k}) m_{k-1}^{(j)}(\mu_{k-1}) , \]  

  (20)

  \[ P_{S,k,k-1}^{(j)}(\mu_{k}, \mu_{k-1}) = F_{S,k,k-1}(\mu_{k}) P_{k-1}^{(j)}(\mu_{k-1}) F_{S,k,k-1}^{T}(\mu_{k}) + Q_{S,k,k-1}(\mu_{k}) . \]  

  (21)

- **Update step**

  For an LGJMS multi-target model, if the predicted intensity \( v_{k,k-1} \) has the form

  \[ v_{k,k-1}(\xi_{k}, \mu_{k}) = \sum_{\mu_{j-1}} \sum_{j=1}^{h_{k,k-1}(\mu_{j-1})} w_{k,k-1}^{(j)}(\mu_{k}) N(\xi_{k}; m_{k,k-1}^{(j)}(\mu_{k}), P_{k,k-1}^{(j)}(\mu_{k})) , \]  

  (22)

  then the posterior intensity \( v_{k} \) at time \( k \) is given by

  \[ v_{k}(\xi_{k}, \mu_{k}) = (1-p_{D,k}(\mu_{k})) v_{k,k-1}(\xi_{k}, \mu_{k}) + \sum_{z \in Z_{k}} v_{D,k}(\xi_{k}, \mu_{k} ; z) , \]  

  (23)

  where \( (1-p_{D,k}(\mu_{k})) v_{k,k-1}(\xi_{k}, \mu_{k}) \) is the mis-detection term, and \( v_{D,k}(\xi_{k}, \mu_{k} ; z) \) refers to the detection terms for each measurement \( z \in Z_{k} \). The calculation expression is as follows:

  \[ v_{D,k}(\xi_{k}, \mu_{k} ; z) = \sum_{j=1}^{h_{k,k}(\mu_{j})} w_{k}^{(j)}(\mu_{k} ; z) N(\xi_{k}; m_{k,k}^{(j)}(\mu_{k} ; z), P_{k,k}^{(j)}(\mu_{k})). \]  

(24)
\[ w_k^{(i)}(\mu_i; z) = \frac{p_{D,k} \left( \mu_k \right) w_{i,k-1}^{(i)}(\mu_k) \eta_k^{(i)}(\mu_i; z)}{\kappa_k(z) + \sum_{\mu_k} p_{D,k} \left( \mu_k \right) \sum_{i} w_{i,k-1}^{(i)}(\mu_k) \eta_k^{(i)}(\mu_i; z)}, \]

(25)

\[ \eta_k^{(i)}(\mu_i; z) = \mathcal{N} \left( z; \mu_k, \sigma_k^{(i)}(\mu_i) \right), \]

(26)

\[ m_{i,k-1}^{(i)}(\mu_i; z) = m_{i,k-1}^{(i)}(\mu_i) + G_k^{(i)}(\mu_i) \left( z - H_k(\mu_i) m_{i,k-1}^{(i)}(\mu_i) \right), \]

(27)

\[ P_{i,k}^{(i)}(\mu_i) = \left( I - G_k^{(i)}(\mu_i) H_k(\mu_i) \right) P_{i,k-1}^{(i)}(\mu_i), \]

(28)

\[ G_k^{(i)}(\mu_i) = P_{i,k-1}^{(i)}(\mu_i) H_k^T(\mu_i) \left( H_k(\mu_i) P_{i,k-1}^{(i)}(\mu_i) H_k^T(\mu_i) + R_k(\mu_i) \right)^{-1}. \]

(29)

Thus, the posterior intensity \( v_k \) at time \( k \) is given by

\[ v_k(\xi_k, \mu_k) = \sum_{i=1}^{l_k} w_k^{(i)}(\mu_k) \mathcal{N}(\xi_k; \xi_k^{(i)}, P_k^{(i)}(\mu_k)). \]

(30)

Given the above, the intensities \( v_{i,k-1} \) and \( v_k \) are analytically propagated in time under the LGJMS multi-target model and the number of Gaussian components of the predicted and posterior intensity increases with time. Therefore, some simple pruning procedures need to be carried out. Finally, the estimate of the multi-target state is the set of ordered pairs of means and modes \( \hat{m}_k^{(i)}(\xi_k, \mu_k) \) with the largest weights \( w_k^{(i)}(\mu_k) \), \( \mu_k \in \mathcal{M}, \ i = 1, \ldots, l_k(\mu_k) \), where \( \hat{N}_k \) is the estimate of the number of targets and equals to \( \sum_{i=1}^{l_k(\mu_k)} w_k^{(i)}(\mu_k) \) rounded to the nearest integer.

3. Deep-Learning-Based Multiple Model Tracking Method

The LGJMS-GMPHD filter described in Section 2 can realize the tracking of multiple maneuvering targets in a cluttered environment. However, there are two key problems in practical applications in relation to targets with complex maneuvering characteristics:

(i) The turn rate parameters of targets are generally unknown and changing, causing a mismatch between the target motion and the filtering model;

(ii) The motion models are varied and switch with uncertainty, and the filter state estimate always lags the current target maneuver, causing the degradation of tracking precision in the maneuvering phase.

Therefore, a deep-learning-based multiple model tracking method is proposed to improve the adaptability of the LGJMS-GMPHD filter to complex maneuvering motions. The relevant flow diagram is shown in Figure 1. Firstly, the multi-target state estimate is obtained by the LGJMS-GMPHD filter. At the same time, track management [27] is performed to obtain the track labels of individual targets. Then, an adaptive turn rate estimation network is designed to realize the real-time estimation of the turn rate, and it is fed back to the filtering process to update the parameters of the CT model, thereby improving the matching degree of the filtering model. On this basis, a filter state modification network is designed to smooth the state estimates. Finally, a trajectory reconstruction is performed to implement the reconstruction of the track segments output by the network, and then the entire target track can be obtained.
Figure 1. Flow diagram of the proposed deep-learning-based multiple model tracking method.

3.1. Adaptive Turn Rate Estimation Network

In JMS, the determination of the turn rate parameter is the key point in the successful application of the CT model. However, in existing methods [15], the turn rate is estimated simply based on the filtering state estimates of the last frame. Additionally, the information dimension is less and the relationship between different variables is not fully utilized, so it is difficult to obtain an accurate estimate of the turn rate.

Deep learning provides an effective means to overcome the limitations of the traditional estimation method [28,29]. A proper network module design can mine more dimensional features, and a diverse set of trajectories can be constructed for network training, which contributes to obtaining more accurate parameter estimates for target trajectories with unknown maneuvering motions. Given all this, we designed an adaptive turn rate estimation network and integrated it into the LGJMS-GMHPD filtering process, thereby improving the tracking precision for CT maneuvering targets. The structure of the network is shown in Figure 2.

Figure 2. Structure of the adaptive turn rate estimation network.
Based on the previous track information from time \( K - w_a + 1 \) to \( K \) obtained by the filtering process, the turn rate estimate at time \( K \) can be obtained by the adaptive turn rate estimation network, where \( w_a \) is the length of the sequence. The length of the sliding window is \( s_a \). First, the feature matrix \( Fv \), including multi-dimensional kinematic information, can be obtained and used as the input of the network. It is represented as follows:

\[
Fv_{K-w_a+1:K} \triangleq \{Fv_{k-w_a+1}, Fv_{k-w_a+2}, \ldots, Fv_k\},
\]

(31)

The feature vector at time \( k \) is represented as follows:

\[
Fv_k = [\hat{x}_k, \hat{y}_k, \hat{x}_k, \hat{y}_k, \hat{r}_k^m, \hat{r}_k^m], \quad k \in [K-w_a+1, K],
\]

(32)

where \( \hat{x}_k \) and \( \hat{y}_k \) are the position estimates, and \( \dot{\hat{x}}_k \) and \( \dot{\hat{y}}_k \) are the velocity estimates.

Meanwhile, based on the track labels, we can also obtain the measurements associated with the track, and then obtain the corresponding Doppler \( r_k^m \), and positions \( x_k^m \) and \( y_k^m \).

The measurement precision of variables in the feature matrix \( Fv \) is different. For example, the target Doppler usually has a higher precision than the position state estimates. Additionally, for each variable, the variation in the temporal dimension reflects the maneuvering characteristics of the target. Therefore, multi-perspective feature extraction is performed to mine more abundant features and improve the accuracy of the turn rate estimates. The temporal dimension feature of each variable is extracted by means of bidirectional long short-term memory (Bi-LSTM).

Then, temporal pattern attention (TPA) [30] is introduced to determine the weights of different variables. The concrete implementation steps are as follows:

- **Bi-LSTM**

  The Bi-LSTM structure is used to capture the temporal correlation information of the feature matrix \( Fv \) at each previous time step. The structure of Bi-LSTM is shown in Figure 3. The output vector \( h_k \) corresponding to the \( k \)-th time point of the \( i \)-th Bi-LSTM is the element-wise sum of the forward and backward LSTM outputs \( \hat{h}_k \) and \( \hat{h}_k \) at the \( k \)-th time point, and is calculated as follows:

\[
h_k = [\hat{h}_k \odot \hat{h}_k].
\]

(33)

Figure 3. Bi-LSTM structure diagram.

In the proposed adaptive turn rate estimate network, two Bi-LSTM layers are integrated, and the output hidden state matrix is represented as \( H^k = \{h_k^1, h_k^2, \ldots, h_k^m\} \), where \( H^k \in \mathbb{R}^{s_a \times m} \), \( m \) is the number of hidden units, and \( h_k^i \) is the hidden state at time step \( k \).

- **Temporal pattern attention**

  For different features in \( Fv \), the temporal patterns across multiple time steps and the weights are extracted by the temporal pattern attention module. First, the \( f \)
A convolutional neural network (CNN) filters $C_i \in \mathbb{R}^{1 \times w}$ are applied on the row vector of $H^t$ to extract temporal pattern features. The convolutional operations yield $H^c_i \in \mathbb{R}^{m \times f}$, where $H^c_{i,j}$ represents the convolutional value of the $i$-th row vector and the $j$-th filter. Formally, this operation is expressed as

$$H^c_{i,j} = \sum_{k=1}^{w} H_{i,k} \times C_{j,k}.$$  

Then, an attention mechanism is carried out. $v_k$ is calculated as a weighted sum of the row vectors of $H^c$ to capture the relational information between different variables. Defined below is the scoring function: $g : \mathbb{R}^k \times \mathbb{R}^m \mapsto \mathbb{R}$ to evaluate relevance:

$$g(H^c, h_k) = (H^c)^T W^c h_k,$$

where $H^c_i$ is the $i$-th row of $H^c$, and $W^c \in \mathbb{R}^{f \times m}$. The attention weight $a_i$ is obtained as

$$a_i = \text{sigmoid}(g(H^c, h_k)).$$

The row vectors of $H^c$ are weighted by $a_i$ to obtain the context vector $v_k \in \mathbb{R}^f$

$$v_k = \sum_{i=1}^{n} a_i H^c_i.$$  

Finally, integrating $v_k$ and $h_k$ to yield the turn rate estimate:

$$h'_k = W^c h_k + W^v v_k,$$

$$\tilde{\omega}_k = W^\omega h'_k,$$

where $h_k, h'_k \in \mathbb{R}^m, W^c \in \mathbb{R}^{m \times f}, W^v \in \mathbb{R}^{f \times m},$ and $W^\omega \in \mathbb{R}^{m \times n}$ is the turn rate estimate.

In the training stage, the root mean square error (RMSE) is used as the loss function:

$$\mathcal{L} = \sqrt{\sum_{k=1}^{K} \left( \tilde{\omega}_k - \omega_k \right)^2},$$

where $\omega_k$ is the true turn rate. The model is trained by minimizing (40) and is optimized through the adaptive moment estimation (Adam) algorithm [31] over the training datasets.

### 3.2. Filter State Modification Network

For strong maneuvering targets, the motion switches between multiple models with uncertainty. The multiple model filtering state estimates have a delay in their response to target maneuvering, resulting in the degradation of the tracking performance. Therefore, a filter state modification network is designed here to smooth the state estimates and advance the tracking precision. The diagram is shown in Figure 4.

The state estimate of each target obtained by the LGJMS-GM-PHD filter is first cropped into a state vector of uniform length $w$, and the length of the sliding window is $s$. It is used as the input of the network and is represented as follows:

$$\tilde{X}_{K-w-1:s} = \left\{ \tilde{X}_{K-w-1+1}, \tilde{X}_{K-w-1+2}, \ldots, \tilde{X}_K \right\},$$
where \( \hat{x}_i = [\hat{x}_k \hat{y}_i]^T, k \in [K-w_t+1, K] \) is the state vector including position estimates.

The variation of the state estimate vector along the temporal dimension can reflect the maneuvering characteristics of the target. Therefore, in the proposed filter state modification network, Bi-LSTM is also applied to extract the temporal correlation information in the forward and reverse directions. In addition, temporal attention (TA) \[32\] is integrated to weight the hidden state vectors at each time step.

Figure 4. Structure of the filter state modification network.

The specific implementation of each module is described as follows.

- **Bi-LSTM**
  
  Two Bi-LSTM layers are integrated to capture the temporal correlation information of state vector \( \hat{x}_{K-w_t+1,K} \) at each previous time step. The output hidden state matrix is represented as \( H^R = \{h_{K-w_t+1}^R, h_{K-w_t+2}^R, \ldots, h_K^R\} \), where \( H^R \in \mathbb{R}^{m \times m} \) is the number of hidden units, and \( h_k^R \) is the hidden state at time step \( k \).

- **Temporal attention**
  
  The hidden state matrix after the temporal attention module is represented as \( H^u = \{h_{K-w_t+1}^u, h_{K-w_t+2}^u, \ldots, h_K^u\} \subseteq \mathbb{R}^{m \times m} \), which is obtained by weighting \( H^R \) at each time step. The weight at time \( k \) is represented as \( \alpha_k^u \) and is calculated as follows:

\[
M_k = \tanh(h_k^u),
\]

\[
\alpha_k^u = \exp(\epsilon_i^u M_i) / \sum_{j=1}^{m} \exp(\epsilon_j^u M_j),
\]

\[
h_k^\alpha = \tanh(\alpha_k^u h_k^u),
\]

where \( \epsilon_i^u \) is a trained parameter vector and the dimension is \( m \). Then \( H^u \) passes through the full connection layer to obtain the modified filter state output:
\[ \tilde{x}_{K-w_i+1:K}^{c'} = W^c H^e + b^c, \]

where \( W^c \) and \( b^c \) represent the weight and bias matrix of the full connection layer, respectively.

In the training stage, the loss function is:

\[ \mathcal{L} = \sqrt{\sum_{i=K-w_i+1}^{K} (\tilde{x}_i^e - x_i^e)^2}, \]

where \( \tilde{x}_i^e \) is the modified filter state estimate and \( x_i^e \) is the ground-truth of the trajectory. The model is trained by minimizing (46) and is optimized through the Adam algorithm over the training datasets.

### 3.3. Trajectory Reconstruction

In the proposed deep-learning-based multiple model tracking method, the historical track information of the target is clipped into segments of appropriate length and used as the input of the deep learning network. Therefore, to obtain the state estimate of the entire trajectory, the modified trajectory segments \( \tilde{x}_{K-w_i+1:K}^{c'} \) output from the filter state modification network need to be connected through a reconstruction step [23].

When the length of a trajectory segment is \( w_i \) and the sliding window is \( s_i \), the overlap region of the adjacent segments is from \( K-w_i+s_i+1 \) to \( K \). The value of the target states in the overlap region is the average of the two adjacent segments, and the state parameters remain constant for non-overlapping regions. The above process is repeated to complete the reconstruction of trajectory segments at each time, and the entire target trajectory can be obtained.

### 4. Network Parameter Design and Performance Analysis

To guarantee the flexibility of the designed deep learning network in real scenarios, a training dataset, including trajectories with different positions, speeds, turn rates, and various maneuvering modes, was constructed. In addition, process noise and measurement noise were added to simulate the target trajectories more realistically. Meanwhile, based on the estimation accuracy of the test dataset, the suitable network structure, network parameters, and input sequence length were designed. Finally, we verified the effectiveness of the two network modules in improving the tracking precision in relation to complex maneuver targets.

For the sake of convenience, in the comparisons of results presented below, LGJMS-GMPHD is abbreviated as GMPHD. After introducing adaptive turn rate estimation, it is denoted as GMPHD-ATN, and after introducing filtering state modification, it is denoted as GMPHD-ATN-FMN.

#### 4.1. Dataset Preparation

4.1.1. Parameter Design

- **Target trajectory parameters**

  We designed five groups of maneuvering trajectories, as shown in Table 1, including different turn rate parameters. A turn rate equal to \( 0^\circ/s \) represents the CV stage and a non-\( 0^\circ/s \) rate represents the CT stage. Meanwhile, two groups of turn rate intervals were designed: \( \omega_1 \in [-8^\circ/s, 8^\circ/s] \) represents a slow turn and \( \omega_2 \in [-15^\circ/s, -8^\circ/s] \cup [8^\circ/s, 15^\circ/s] \) represents a fast turn. In addition, the durations of different motion stages are different. Specifically, CV is 20–25 s, \( \omega_1 \) is 10–50 s, and \( \omega_2 \) is 8–20 s. For each group of trajectories
in the training datasets, the kinematic parameters were uniformly extracted according to Table 1.

Table 1. The kinematic parameters of the target trajectory.

<table>
<thead>
<tr>
<th>Group Index</th>
<th>Initial Velocity (m/s)</th>
<th>Turn Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Stage 1</td>
</tr>
<tr>
<td>1</td>
<td>[−60, 60]</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>[−40, 40]</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>[−60, 60]</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>[−60, 60]</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>[−40, 40]</td>
<td>0</td>
</tr>
</tbody>
</table>

- Filter parameters
  For the LGJMS-GMPHD filter process, $\mu=1$ is the CV model and $\mu=2$ is the CT model. The switching between the motion models is given by the Markov transition probability matrix as

$$
\mathbf{T}_{k-1} = \begin{bmatrix}
0.9 & 0.1 \\
0.1 & 0.9
\end{bmatrix}, \quad (47)
$$

For simplicity, we assume that the probability of target survival and detection is constant. $p_{s,k-1} = 0.99$ and $p_{d,k} = 0.98$ are assumed for models $\mu = 1, 2$. Additionally, the interval between the samples is $T = 1 \text{ s}$. The standard deviation of process noise $\sigma_v = 1 \text{ m/s}^2$ and the standard deviation of measurement noise $\sigma_r = 10 \text{ m}$.

4.1.2. Dataset Construction
- Training and validation datasets
  The construction of the datasets for the two networks is summarized in the flowchart shown in Figure 5. The details are described as follows:
  (a) The kinematic parameters from Table 1 are randomly selected to generate real target trajectories.
  (b) Measurement noise is added and target measurements are obtained.
  (c) The LGJMS-GMPHD filter process is performed to obtain the state estimate, where the turn rate of CT model is calculated via a physical definition method to obtain the dataset of the adaptive turn rate estimate network, and the true value is used to obtain the datasets of the filter state modification network.
  (d) The trajectories are clipped to segments of uniform length. For the adaptive turn rate estimation network, the feature matrix $F_v$ defined in (31) is stored. For the filter state modification network, the position state estimate $\hat{x}_c$ defined in (41) is stored.
  (e) Data normalization is performed based on min-max scaling.
  (f) The order is disrupted and the training and validation datasets are divided.

  In this paper, the number of trajectory segments in the training and validation datasets for the two networks was 300,000 and 60,000, respectively. In the training procedure, the learning rate was set to $10^{-4}$, the batch size was 100 samples, and the training epoch was 300.
Figure 5. Flowchart of dataset construction.

- Test datasets

  Based on the trajectory parameters shown in Table 1, 100 maneuvering trajectories were constructed for each group to verify the effectiveness of each module in the proposed method in improving tracking performance.

4.2. Network Module Parameter Design

4.2.1. Adaptive Turn Rate Estimation Network

The network structures, parameters, and input sequence length were validated based on the five groups' trajectories with different maneuvering forms in the test datasets. The evaluation criterion is the RMSE of turn rate estimate and defined as follows:

\[
\sigma_\omega(k) = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (\hat{\omega}(k) - \omega(k))^2}
\]  

(48)

where \( \hat{\omega}(k) \) is the output of the adaptive turn rate estimation network, and \( \omega(k) \) is the true value. \( N \) is the number of simulations.

- Network structure

  The number of Bi-LSTM layers and the module effectiveness were verified based on the RMSE of the turn rate estimate, as shown in Table 2. Here, we set the number of hidden units of the Bi-LSTM to 64, the sequence length to 10, and the sliding window to 1. Compared to one Bi-LSTM layer, two Bi-LSTM layers can produce a higher accuracy, but the improvement is not significant as the number of layers continues to increase. Therefore, we set the number of Bi-LSTM layers as two. After the TPA module was introduced, it can be seen from the last column that the estimation error was greatly reduced. Especially for the second group, the RMSE was reduced by 0.71°, which verifies the effectiveness of the TPA module.

<table>
<thead>
<tr>
<th>Group Index</th>
<th>Network Structure</th>
<th>One Bi-LSTM Layer</th>
<th>Two Bi-LSTM Layers</th>
<th>Three Bi-LSTM Layers</th>
<th>Four Bi-LSTM Layers</th>
<th>Two Bi-LSTM Layers + TPA</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>One Bi-LSTM Layer</td>
<td>0.90</td>
<td>0.76</td>
<td>0.76</td>
<td>0.77</td>
<td>0.55</td>
</tr>
<tr>
<td>2</td>
<td>Two Bi-LSTM Layers</td>
<td>2.06</td>
<td>1.63</td>
<td>1.56</td>
<td>1.45</td>
<td>0.92</td>
</tr>
</tbody>
</table>
• Network parameters

After the network structure was determined, we analyzed the influence of the number of hidden units of the Bi-LSTM layers on the performance. The RMSE of the turn rate estimates with different numbers of hidden units for the Bi-LSTM layers is shown in Table 3. Here, we set the sequence length to 10 and the sliding window to one. As the number of hidden units increased, the RMSE decreased, but the reduction was not significant for 64, 96, and 128. Therefore, considering operational efficiency and estimation precision comprehensively, we set the number of hidden units to 64.

Table 3. RMSE of different network parameters (units: degree).

<table>
<thead>
<tr>
<th>Group Index</th>
<th>Number of Hidden Units</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>32</td>
</tr>
<tr>
<td>1</td>
<td>0.62</td>
</tr>
<tr>
<td>2</td>
<td>1.06</td>
</tr>
<tr>
<td>3</td>
<td>0.82</td>
</tr>
<tr>
<td>4</td>
<td>0.97</td>
</tr>
<tr>
<td>5</td>
<td>1.09</td>
</tr>
</tbody>
</table>

• Sequence length

Long sequences generally contain more abundant feature information, but this also degrades the real-time performance and increases the difficulty of network training. Therefore, we analyzed the influence of input sequence length on the estimate precision of the network. Here, we set the number of hidden units to 64 and the sliding window to one. The RMSE comparisons for different sequence length are shown in Table 4. When the sequence length increased from 6 to 12, the RMSE of the turn rate estimate decreased gradually. However, when the length was 14, the RMSE increased slightly compared with 12. Therefore, we set the input sequence length of the turn rate estimation network as 12.

Table 4. RMSE of different input sequence lengths (units: degree).

<table>
<thead>
<tr>
<th>Group Index</th>
<th>Sequence Length</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>6</td>
</tr>
<tr>
<td>1</td>
<td>0.64</td>
</tr>
<tr>
<td>2</td>
<td>0.98</td>
</tr>
<tr>
<td>3</td>
<td>0.80</td>
</tr>
<tr>
<td>4</td>
<td>1.04</td>
</tr>
<tr>
<td>5</td>
<td>1.13</td>
</tr>
</tbody>
</table>

4.2.2. Filter State Modification Network

The network structure, parameters, and input sequence length were validated based on the testing datasets. The evaluation criterion was the RMSE and was defined as follows:

\[
\sigma_M(k) = \frac{1}{N} \sum_{i=1}^{N} \left( \left( \hat{x}_i - x_i \right)^2 + \left( \hat{y}_i - y_i \right)^2 \right)
\]  \hspace{1cm} (49)

where \( \hat{x}_i \) and \( \hat{y}_i \) represent the output of the network, and \( x_i \) and \( y_i \) are the real values; \( N \) is the number of simulations.

• Network structure
Comparisons of the RMSE of different network structures are shown in Table 5. Here, we set the number of hidden units to 64, the length of input sequence to 15, and the sliding window to five. First, when the number of Bi-LSTM layers increased from one to two, the output precision of the network was obviously improved. However, when the number of layers increased to three and four, the RMSE of groups 1, 3, and 4 increased instead; that is, the network was overfitted. Therefore, we set the number of Bi-LSTM layers to two. After the temporal attention module was introduced, as shown in the last column in Table 5, the RSME was further reduced. Especially for groups 1, 3, and 4, the reduction was remarkable, which verifies the effectiveness of the TA module.

Table 5. RMSE of different network structures (units: m).

<table>
<thead>
<tr>
<th>Group Index</th>
<th>Network Structure</th>
<th>One Bi-LSTM Layer</th>
<th>Two Bi-LSTM Layers</th>
<th>Three Bi-LSTM Layers</th>
<th>Four Bi-LSTM Layers</th>
<th>Two Bi-LSTM Layers + TA</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>11.96</td>
<td>11.62</td>
<td>12.52</td>
<td>13.13</td>
<td>11.28</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>10.23</td>
<td>9.93</td>
<td>9.99</td>
<td>10.02</td>
<td>9.88</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>11.61</td>
<td>11.20</td>
<td>12.68</td>
<td>13.07</td>
<td>10.66</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>11.71</td>
<td>11.21</td>
<td>12.91</td>
<td>13.43</td>
<td>10.84</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>11.41</td>
<td>10.86</td>
<td>10.98</td>
<td>10.99</td>
<td>10.82</td>
</tr>
</tbody>
</table>

- **Network parameters**

  The RMSE of test datasets for different numbers of hidden units is shown in Table 6. Here, we set the length of the input sequence to 15 and the sliding window to five. First, with the increase in the number of hidden units, the network performance was improved, but the increase was not obvious. The RMSE for 64 hidden units was reduced by about 0.2 m compared with that for 32 hidden units. Compared with 64, the performance improvement for 96 and 128 hidden units was about 0.1 m. Meanwhile, for groups 1–4, overfitting also occurred when the parameter was 128. Hence, we set the number of hidden units to 64.

Table 6. RMSE of different network parameters (units: m).

<table>
<thead>
<tr>
<th>Group Index</th>
<th>Number of Hidden Units</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>32</td>
</tr>
<tr>
<td>1</td>
<td>11.46</td>
</tr>
<tr>
<td>2</td>
<td>9.91</td>
</tr>
<tr>
<td>3</td>
<td>10.81</td>
</tr>
<tr>
<td>4</td>
<td>11.01</td>
</tr>
<tr>
<td>5</td>
<td>10.85</td>
</tr>
</tbody>
</table>

- **Sequence length**

  The influence of the input sequence length on the precision of the filter state modification network was analyzed. Comparisons of the RMSE of the testing datasets are shown in Table 7. Here, we set the number of hidden units to 64 and the sliding window to five. Horizontally, for each group of test trajectories, the RMSE decreased with the increase in the sequence length. However, when the sequence length was 15, the error was at its minimum, and when the sequence length increased to 16, the performance slightly decreased. Therefore, we set the input sequence length to 15.
Table 7. RMSE of different input sequence lengths (units: m).

<table>
<thead>
<tr>
<th>Group Index</th>
<th>Sequence Length</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>8</td>
</tr>
<tr>
<td>1</td>
<td>12.29</td>
</tr>
<tr>
<td>2</td>
<td>10.37</td>
</tr>
<tr>
<td>3</td>
<td>11.67</td>
</tr>
<tr>
<td>4</td>
<td>12.16</td>
</tr>
<tr>
<td>5</td>
<td>11.36</td>
</tr>
</tbody>
</table>

4.3. Network Module Performance Validation

In this section, five groups of target trajectories with different maneuvering modes in the testing datasets were extracted to verify the validity of the two network modules in the proposed deep-learning-based multiple model tracking method. To show the result comparisons more intuitively, we extracted a target trajectory from each group of maneuvering modes, and the comparison of single simulation results is shown in Figure 6. The first column represents the comparison of the tracking results, the second column represents the comparison of the turn rate estimation results, and the third column represents the comparison of the tracking errors.
For the traditional physical definition method, only the filtering state estimates in the last frame were used. In the maneuvering phase, the estimation error of the turn rate was large and fluctuated greatly, which led to a decrease in the tracking precision. Especially for the strong maneuvering trajectory with a large turn rate, for example, the second target performed the turning motion with the parameter $\omega = -13^\circ/s$ between frames 20 and 38, and the tracking RMSE in the corresponding period increased rapidly. For the proposed adaptive turn rate estimation network, multi-frame kinematic features can be utilized and more complex logical relationships between different variables can be extracted, which contributes to obtaining more accurate estimates of turn rates, as shown by the red curve in Figure 6b. Although the estimation error in the maneuvering phase was slightly larger than that in the non-maneuvering phase, the performance gradually became stable after several frames. The corresponding tracking results are indicated by the red curve in Figure 6c. Compared with the GMPHD filter, our method can effectively reduce the tracking error in the turn maneuvering phase.

As can be seen from the tracking results in Figure 6a and the tracking error curve in Figure 6c, an accurate estimate of the turn rate can improve the tracking performance to a certain extent. However, there is uncertainty in the switching of target motion models, resulting in large tracking errors. The filter state modification network can smooth the target state estimates that are produced by the GMPHD filter. As shown in Figure 6a, the tracking result of the purple curve is smoother than those of the red and green curves. Additionally, it can also be seen in Figure 6c that the RMSE corresponding to the blue curve greatly decreased during the turning maneuver phase.

Furthermore, from the tracking result comparisons of all trajectories in the testing datasets shown in Figure 7, it can be concluded that the proposed method has strong adaptability to multiple maneuvering modes and can effectively enhance the tracking precision of complex maneuvering targets.
5. Simulations and Experimental Results

In this section, the adaptability of the proposed method to the multiple maneuvering target tracking in a cluttered environment is demonstrated.

5.1. Simulations

5.1.1. Simulation Scenario and Parameters

Multiple maneuvering targets in a cluttered environment were constructed to verify the tracking performance of the proposed method. The filter parameter design was the same as that presented in Section 4.1.1. In addition, the detected measurements were immersed in clutter, which can be modeled as a Poisson RFS $K_z$ with intensity

$$\kappa(z) = \lambda V U(z)$$

(50)

where $U(\cdot)$ is the uniform density over the surveillance region, $V = 1.6 \times 10^7 m^2$ is the ‘volume’ of the surveillance region, and $\lambda \approx 3.2 \times 10^{-6} m^{-2}$ is the average number of clutter returns per unit volume.

We designed five target trajectories, representing the five maneuvering forms shown in Table 1 respectively, and the trajectory parameters of each target are shown in Table 8. The simulation scenario is shown in Figure 8.

Table 8. Target trajectory parameters.

<table>
<thead>
<tr>
<th>Index</th>
<th>Initial State</th>
<th>Turn Rate (°/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>[500 m; 20 m/s; 523 m; -25 m/s]</td>
<td>1-24 s</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>-7</td>
</tr>
<tr>
<td>2</td>
<td>[243 m; -35 m/s; 266 m; -20.4 m/s]</td>
<td>1-21 s</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>-10.4</td>
</tr>
<tr>
<td>3</td>
<td>[941 m; -27.4 m/s; 737 m; 35.5 m/s]</td>
<td>1-21 s</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>5.2</td>
</tr>
<tr>
<td>4</td>
<td>[1000 m; 15 m/s; 1300 m; -20 m/s]</td>
<td>1-25 s</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>2.9</td>
</tr>
<tr>
<td>5</td>
<td>[109 m; -30 m/s; 620 m; -30 m/s]</td>
<td>1-23 s</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>13.7</td>
</tr>
</tbody>
</table>
5.1.2. Result Comparisons

Based on the simulation scenarios shown in Figure 8, the tracking performance of the proposed method was verified, and the optimal subpattern assignment (OSPA) metric [33] was adopted to evaluate the performance of multi-target tracking. After 100 Monte Carlo simulation runs, the OSPA distance comparison is shown in Figure 9, and the turn rate estimate error is shown in Figure 10.

First, from 25–32 s and 71–82 s, targets 2, 3, and 5 all contained fast turning maneuvers with the parameter $\omega$, and the OSPA distance of the GMPHD filter increased dramatically. For GMPHD-ATN, it did not show a large fluctuation, and compared with the

---

**Figure 8.** Simulation scenario. (△—start position, ◯—stop position).

**Figure 9.** OSPA distance comparison.

**Figure 10.** RMSE of turn rate estimates.
GMPHD filter, the maximum reduction was about 12 m, which verifies the importance of turn rate estimate accuracy for the stable tracking of strong maneuvering targets. Additionally, for the GMPHD-ATN-FMN, the introduction of the filter state modification network module was able to smooth the state estimate. Therefore, the tracking precision was further significantly improved in the entire tracking stage, and the performance was improved significantly in the strong maneuvering phase. Specifically, in the weak maneuvering stage, such as 35–70 s, the maximum reduction of OSPA distance was about 4 m, whereas in the strong maneuvering stage, such as 25–32 s, the maximum reduction was about 7.5 m.

Figure 11 shows comparisons of the tracking results obtained in a single simulation. The estimate results of the turn rate for the five targets are shown in Figure 12. For target 5, the large estimate error of the turn rate in GMPHD led to a mismatch in the filtering parameters, and the track was broken in area 3. After introducing the adaptive turn rate estimation process, as shown in Figure 12, more accurate estimates of the turn rate can be obtained. Although the estimate error increased slightly when the motion model switched, after several frames the error gradually decreased. As indicated by the green curve in Figure 11, the matched filtering parameters were able to effectively improve the tracking precision and avoid track breakages in the maneuvering stage. Additionally, as shown by the purple curve in Figure 11, the filter state modification process can smooth the trajectory and output higher quality maneuvering target tracks.

![Figure 11. Tracking results of a single simulation.](image-url)
Figure 12. Result comparisons of turn rate estimate: (a) target 1, (b) target 2, (c) target 3, (d) target 4, and (e) target 5.

5.2. Experimental Data Validation

5.2.1. Experimental Scenarios and Parameters

In this section, the tracking performance of the proposed method is further verified based on experimental field data. The experimental equipment is a phased array radar working in the Ku band. The sample interval is 6 s. In the experiment, the flight height of a small UAV (DJI Matrice 600) is approximately 200 m with CT maneuvers. The experimental scenario is shown in Figure 13.

Figure 13. Experimental scenario.

5.2.2. Result Comparisons

After the echo data collected by the radar were processed by constant false alarm rate detection, measurements of the area where the UAV was located could be obtained. To verify the tracking performance in the multiple targets scenario, we combined two sets of measurements of the single UAV target into one set.

Figure 14 shows the tracking result comparisons. Firstly, for the GMPHD with unknown turn rate parameters, the mismatch in the model parameter resulted in track breakages at the turning position. For the GMPHD-ATN, the track quality was significantly improved. Additionally, as shown in Figure 14c, after introducing filter state modification processing, the tracking result was smoothed, and the precision of the state estimate was effectively improved. Therefore, the tracking results of real UAV observation data verify the rationality of the parameters of the constructed training datasets, and the proposed method demonstrated its adaptability to practical scenarios with detection and measurement noise uncertainties.
Figure 14. Comparisons of real UAV tracking results: (a) GMPHD, (b) GMPHD-ATN, and (c) GMPHD-ATN-FMN.

5.2.3. Computational Complexity

For a fair comparison, the computational efficiency of all algorithms was tested on an Intel Xeon E5-2680 CPU at 2.4 GHz. For the adaptive turn rate estimation network, the testing runtime per iteration was 2 ms, and for the filter state modification network, the testing runtime per iteration was 4 ms. Although the training of the deep neural network was time consuming, its implementation in practice is highly efficient because the calculations are mainly matrix multiplications and element-wise operations. Therefore, the proposed method can ensure the real-time performance of the tracking process.

6. Discussion

The RFS-based filter incorporating the jump Markov system approach is an effective method for multiple maneuvering target tracking in cluttered environments. However, UAV targets have strong maneuverability and their trajectory forms are diverse, presenting unknown motion model parameters and uncertain model switching. The preset parameters of filter models are difficult to match to the time-varying target motion, leading to a serious decline in tracking performance. Therefore, within the framework of the LGJMS-GMPHD filter, we proposed a deep-learning-based multiple model tracking method to improve maneuvering adaptability.

The simulation results indicated that the traditional LGJMS-GMPHD filter has a low tracking precision and the track is prone to breaking in scenarios involving complex maneuvering targets. As shown in Figure 12, for the PDM, the estimation error of the turn rate was large and fluctuated greatly in the maneuvering phase as only the filtering state estimate in the last frame was considered, whereas the proposed adaptive turn rate estimation network made use of multiple dimensions of kinematic features and extracted the relationship between different features. As shown in the red curve in Figure 12, a more accurate estimation of turn rate can be obtained, which helps to improve the matching degree of the CT model in the filter. Therefore, as shown in Figure 9, compared with the GMPHD filter, the OSPA distance of GMPHD-ATN was greatly reduced, and the tracking results shown in Figure 11 indicate that track continuity was optimized.

Another reason for poor tracking performance is the uncertainty of motion model switching. Smoothing results in better estimates for target states by means of a time delay. However, as described in [21], longer lags do not yield better estimates of the current state because the temporal correlations cannot be extracted. Moreover, backward smoothing is still model-based, which cannot address the substantive problem of tracking performance degradations due to model switching. Deep neural networks have a strong capability of fitting any mapping, providing an effective way to handle target motion uncertainty. However, the LSTM-based deep recurrent neural network in [24] only considered CV and CA motions, and is not suitable for the scenario of multiple maneuvering targets in the CT model. However, in the proposed filter state modification network, the state estimates
output by the LGJMS-GMPHD filter were used as the input, and the temporal correlation information of the state vector was captured, with the ability to handle complex maneuvering motions. Hence, as shown in Figure 9, the tracking precision was improved in the entire tracking stage, especially in the strong maneuvering phase. The tracking results, shown in Figure 11, demonstrated that the track was smoothed and the track quality was improved. Moreover, based on the tracking error comparisons of test datasets, shown in Figure 7, it can be concluded that the proposed method has strong adaptability to different maneuvering forms. The experimental data processing results indicate that the parameters of the training dataset designed in this paper can adapt to practical scenarios, and the method can effectively enhance the track quality of real UAV targets.

However, in this paper, we assumed that the clutter rate was known and remained constant. In future research, more complex detection environments, such as a time-varying clutter rate and detection uncertainty, will be considered. Furthermore, we will also consider extending the proposed method to group target tracking.

7. Conclusions

In this paper, we have proposed a deep-learning-based multiple model tracking method. The adaptive turn rate estimation network employs multi-frame and multi-dimensional kinematic information to improve the accuracy of the turn rate estimates for a maneuvering target with CT motion, thereby enhancing the consistency with the filtering model. Additionally, the filter state modification network uses the temporal features of the multi-frame state estimate to achieve the smoothing of target state estimates, thereby circumventing the large tracking errors caused by uncertain motion model switching. To ensure the applicability of the algorithm, in the training datasets, we designed five switching modes of the movement model. Therefore, the proposed method is suitable for practical maneuvering movements: (i) from CV to CT, (ii) from CT with \( \omega_1 \) to CT with \( \omega_2 \), and (iii) from CT to CV. In the end, based on the simulation results of multiple maneuvering target tracking in cluttered environments, it can be concluded that the proposed method can obtain accurate turn rate estimates and output high-quality tracks, and the performance improvement is especially significant in the maneuvering phase. Moreover, for the real UAV observation scenario with measurement noise and detection uncertainties, the proposed method can output more stable and smooth trajectories, which verifies its applicability in real scenarios.

Author Contributions: Conceptualization, L.F. and W.T.; methodology, L.F. and W.T.; software, L.F.; validation, L.F. and R.W.; formal analysis, R.W. and H.L.; investigation, R.W.; resources, W.L. (Weigang Luo), N.N., and H.L.; data curation, H.L. and W.L. (Weidong Li); writing—original draft preparation, L.F. and W.T.; writing—review and editing, R.W. and L.F.; visualization, N.N. and W.L. (Weidong Li); supervision, C.H.; project administration, C.H.; funding acquisition, C.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China (Grant No. U2133217), the National Natural Science Foundation of China (Grant No. 31727901) and the National Natural Science Foundation of China (Grant No. 62001021).

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References


