Hindawi
International Journal of Aerospace Engineering
Volume 2022, Article ID 8220029, 16 pages
https://doi.org/10.1155/2022/8220029

Research Article

Combined AGADESN with DBSCAN Algorithm for Cluster Target Motion Intention Recognition

Xirui Xue, Shucai Huang, and Daozhi Wei

Air and Missile Defense College, Air Force Engineering University, Xi'an 710051, China

Correspondence should be addressed to Xirui Xue; rayngu@126.com

Received 10 July 2022; Accepted 7 October 2022; Published 22 October 2022

Academic Editor: Jinchao Chen

In this paper, we consider the problem of motion intention recognition for cluster targets with splitting behaviour and lack of motion prior information. This is a challenge to the classical Bayesian inference based intention recognition algorithms because they rely heavily on a priori knowledge. In order to solve these problems, a joint algorithm of deep echo state network optimized by adaptive genetic algorithm (AGADESN) and DBSCAN clustering algorithm is proposed in this paper. We use improved Olfati-Saber model with direction noise to generate cluster motion and use the cluster motion data to drive AGADESN algorithm to predict cluster destination, which achieves higher destination prediction accuracy than DESN algorithm. We innovatively design the motion similarity distance (MSD) and take the destination prediction output as one of the distance inputs, alleviating the lack of differentiation among different cluster targets caused by only relying on speed and position distance at the early stage of cluster motion. Based on the MSD, DBSCAN clustering algorithm is used to identify clusters in the field of view to determine whether splitting behaviour occurs. Simulation results demonstrate the effectiveness of the proposed algorithm in cluster target motion intention recognition and its superiority over DESN algorithm and DBSCAN algorithm only based on speed and position distance.

1. Introduction

In recent years, the recognition and prediction of the moving intention of cluster targets have attracted extensive attention. Cluster target intention recognition and prediction refers to the automatic evaluation and prediction of a certain goal or plan to be achieved by the cluster target in the region. It is an important function of situation analysis and belongs to the high-level processing part of the data fusion system [1]. At present, UAV cluster has been used in environmental detection, target tracking, and other aspects and has also been widely used in several recent regional conflicts. Solving the problem of cluster intention recognition would help better solve the problems of UAV cluster control [2, 3], target tracking [4], path planning [5, 6], etc. In particular, in UAV path planning, it is very important to effectively infer the intention of UAV clusters to solve the coverage path planning problem [7, 8]. However, due to the difficulty of modeling cluster trajectory, the lack of prior information about the motion and the existence of multiple motion patterns such as splitting and merging, it is difficult to recognize cluster motion intention. In addition, some clusters have strong autonomous ability, obvious heterogeneous characteristics and automatic path planning ability, which brings more difficulties to the recognition of cluster motion intention [9, 10].

In order to solve these problems, it is necessary to establish a reasonable collaborative motion model of cluster targets. A pioneering study by Reynolds [11] demonstrated that cluster motion follows three basic collaborative rules: separation, speed matching, and aggregation. A similar approach is proposed by Vicsek et al. [12], who believed that the collaborative motion comes from adjusting the speed of cluster members to the average speed in their neighbourhood. Other studies have focused on the dynamic behaviour of clusters. For example, Couzin et al. [13] attempted to explain the clustering phenomenon under effective guidance and group decision-making. However, clusters described only by three basic collaborative rules are prone to fracture due to the dispersion of member positions. On the basis of the basic rules, Olfati-Saber [14] designed and added the virtual leader in order to solve the problem of cluster fracture. However, the assumption in the basic Olfati-Saber model that the virtual leader information is accurate all the time is too idealistic. Considering the difficulty of virtual leader information acquisition in the actual cluster movement process, Gaussian noise with time-varying variance is introduced to make the cluster cooperative motion model more realistic.

Traditional cluster target intention recognition mainly relies on Bayesian network inference framework [15]. Bayesian reasoning has the advantages of dealing with the inherent uncertainty of sensor data and the inherent semantic fuzziness of intention recognition, supporting the relearning of reasoning network parameters [16, 17]. In order to overcome the problem of aircraft carrier group target intention reasoning in multidomain operations and multientity hierarchical, Bayesian network is used by Qiao et al. [18]. However, model-driven Bayesian reasoning inevitably needs to use the prior statistical information of the target, which is usually difficult for us to obtain. Especially in the military field, the random, sudden, and unknown maneuver of enemy UAV cluster targets further increase the difficulty of obtaining prior information.

In recent decades, sensor and computer technology have promoted the rapid development of big data processing. In particular, deep learning and neural networks have made many impressive advances and have excellent performance in intention recognition [19, 20]. In order to recognize and predict the motion intention of cluster targets, the data-driven neural network method needs to use the target motion sequence information. Echo state network (ESN) is widely used in time series prediction because of its simple structure and efficient training [21]. However, the motion sequence information of cluster targets is more complex, and the prediction ability of ESN is insufficient. Accordingly, some scholars have proposed a deep echo state network (DESN) model [22, 23].

In this paper, we divide cluster target intention recognition into two parts: destination prediction and cluster identification. Song et al. [24] used DESN to predict taxi destination. However, in order to achieve the lowest learning error, arbitrarily set network parameters may not guarantee the best training results of DESN. Chen and Zhang and Lu et al. [25, 26], respectively, used grey wolf algorithm and genetic algorithm (GA) to optimize DESN parameters. However, in the process of application, we find that DESN optimized by GA converges slowly, and the parameter optimization result after convergence is not ideal. Accordingly, we further develop the optimization process of GA and propose a genetic algorithm with adaptive mutation rate [27] to optimize DESN.

In terms of cluster identification, Snidaro et al.[28] used the evidence reasoning algorithm in D-S evidence theory, with the help of prior knowledge and multisensor detection information, to complete the identifying and clustering of targets. Oxenham et al. [29] extracted the situation elements of Bayesian network and used data mining to achieve target clustering. However, due to the small size of the cluster target members, under the existing detection conditions, the members attribute information is difficult to obtain. Moreover, the cluster motion prior information is difficult to obtain, so traditional clustering algorithm is still one of the most effective cluster identification algorithms.

Clustering algorithms mainly include data-based clustering algorithms and parameter-likelihood-based clustering algorithms [30]. Data-based clustering algorithms include hierarchical clustering [31] and partitioned clustering such as k-means estimator [32]. Density-based noise application spatial clustering (DBSCAN) is a typical data-based clustering algorithm [33, 34], which can be used to solve the clustering problem with uncertain data. However, when the distance between target members is close, the performance of DBSCAN algorithm decreases. This paper defines a new distance measure based on destination prediction, which is called motion similarity distance (MSD). At this distance, DBSCAN algorithm is applied to cluster identification, and a good clustering effect is achieved.

A joint algorithm of AGADESN and DBSCAN is proposed to solve the problem of cluster target motion intention recognition under the condition of lack of motion prior information and cluster splitting. The main contributions of this paper are described as follows:

(1) The direction noise with time-varying variance is introduced to improve the virtual leader. Under the framework of Olfati-Saber model, a more practical cluster target collaborative motion model is proposed. The proposed algorithm takes into account the source of command information when the cluster is moving, describes the process of gradually reducing the noise in the moving direction as the cluster approaches the destination, and considers the compensation of the cluster speed for the noise interference

(2) The adaptive mutation rate is introduced into the genetic algorithm, and the adaptive genetic algorithm with faster iteration speed is used to optimize the parameters of the DESN. The AGADESN algorithm is proposed

(3) Based on the cluster motion model proposed in this paper, the cluster motion trajectory dataset for training and test is established, and under the sliding window structure, the cluster motion destination is predicted by using AGADESN model. Experimental results show that our algorithm improves the correct prediction ratio of basic DESN algorithm by 8.39%.

(4) A new distance measure MSD is proposed, which combines the predicted destination distance with the position distance and speed distance, and the DBSCAN algorithm is used to identify the cluster to judge the splitting behaviour on the MSD. The experimental results show that the DBSCAN algorithm using MSD can quickly identify the clusters in the sensor field of view and has stronger robustness when the number of clusters changes.

The rest of this paper is organized as follows. In Section 2, improved Olfati-Saber is used to describe the cluster collaborative motion. In Section 3, we introduce the joint algorithm of motion intention recognition. Section 4 analyses the generated cluster motion trajectory and gives the results and discussion of the motion intention recognition simulation experiment. Section 5 summarizes the paper.

2. Cluster Collaborative Motion Model

2.1. Fundamentals of the Olfati-Saber Model. The Olfati-Saber cluster collaborative motion model adds a virtual leader on the basis of the three cluster collaborative motion principles of separation, speed matching and aggregation. All members of the cluster can obtain the position and speed information of the virtual leader. After simplifying the cluster members into the particle model, at time t, the Olfati-Saber model describes the motion of member i as follows:

$$\begin{aligned}\dot{\mathbf{q}}_i(t) &= \mathbf{p}_i(t),\\ \dot{\mathbf{p}}_i(t) &= \mathbf{u}_i(t),\end{aligned} \tag{1}$$

where $\mathbf{q}_i(t)$ and $\mathbf{p}_i(t)$ are the position and speed vectors of member i at time t, respectively. $\mathbf{u}_i(t)$ is the control input of member i at time t. According to the principle of cluster collaborative motion proposed by Olfati-Saber model, $\mathbf{u}_i(t)$ is designed as follows:

$$\mathbf{u}_i(t) = \mathbf{f}_i^s + \mathbf{f}_i^m + \mathbf{f}_i^d, \tag{2}$$

where $\mathbf{f}_i^s$ is the separation and aggregation control input, which is used to adjust the spacing between members to avoid collisions between members. $\mathbf{f}_i^m$ is the speed matching control input, which ensures the information transmission within the cluster, makes the cluster topology have weak connectivity, and realizes the gradual consistency of the members' speed. $\mathbf{f}_i^d$ is the control input of the virtual leader, which is used to plan the overall motion path of the cluster.

The description of $\mathbf{f}_i^s$ is shown as follows:

$$\mathbf{f}_i^s = -\sum_{j\in N_i(t)} \nabla_{\mathbf{q}_i} \Psi_\alpha\left(\left\|\mathbf{q}_j - \mathbf{q}_i\right\|_\sigma\right), \tag{3}$$

where $\|\cdot\|_\sigma$ represents the σ-norm, and its calculation method is shown as follows:

$$\|z\|_\sigma = \frac{1}{\varepsilon}\left[\sqrt{1+\varepsilon\|z\|^2} - 1\right]. \tag{4}$$

Here, the fixed parameter is $\varepsilon \in (0, 1)$.

$\|\mathbf{q}_j - \mathbf{q}_i\|_\sigma$ denotes the σ-norm distance between member i and member j. $N_i(t)$ represents the set of members in the neighbourhood of member i that can interact with member i. Neighborhood can be selected according to different metrics, and widely used distance metrics include Euclidean distance and topological distance [35, 36].

$\Psi_\alpha(\cdot)$ represents the potential energy function, which needs to meet the requirements of continuity, differentiability, and nonnegativity, and its magnitude is related to the distance between the two members. $\nabla\Psi_\alpha(\cdot)$ represents the gradient of the potential energy function, so as to convert the potential energy into the separation and aggregation control input.

The form of $\Psi_\alpha(\cdot)$ is defined as follows:

$$\Psi_\alpha(z) = \int_{d_\alpha}^{z} \Phi_\alpha(s)\mathrm{d}s, \tag{5}$$

where d is the expected distance between members after the member speed converges. d_α is the transformation form of σ-norm distance of d and $d_\alpha = \|d\|_\sigma$.

$\Phi_a(\cdot)$ is the potential force function between members. When the distance between member i and member j is less than the distance threshold, there is repulsive force between them to prevent collision. Otherwise, there is an attractive force to promote the cluster aggregation. The expression of $\Phi_a(\cdot)$ is as follows:

$$\Phi_\alpha(z) = \rho_h(z/r_\alpha)\Phi(z - d_\alpha), \tag{6}$$

where r_α is the finite cut-off point, and $r_\alpha = \|r\|_\sigma$. r represents the maximum distance that information interaction can occur between members. $\rho_h(z)$ is the smooth scalar function with a value range of [0, 1] [37]. The introduction of $\rho_h(z)$ makes the potential force have limited extreme values and makes the spatial adjacency matrix smoother, which is defined as follows:

$$\rho_h(z) = \begin{cases} 1 & z \in [0, h), \\ \dfrac{1}{2}\left[1 + \cos\left(\pi\dfrac{z-h}{1-h}\right)\right] & z \in [h, 1], \\ 0 & \text{otherwise}, \end{cases} \tag{7}$$

$\Phi(z)$ is a nonuniform s-shaped function, expressed as follows:

$$\Phi(z) = \frac{1}{2}\left[(a+b)\left(\frac{z+c}{\sqrt{1+(z+c)^2}}\right) + (a-b)\right]. \tag{8}$$

To ensure $\Phi(0) = 0$, the parameters in the Equation (8) need to meet $0 < a \le b$ and $c = |a-b|/\sqrt{4ab}$.

The description of $\mathbf{f}_i^m$ is shown as follows:

$$\mathbf{f}_i^m = \sum_{j\in N_i(t)} a_{ij}(\mathbf{q})\left(\mathbf{p}_j - \mathbf{p}_i\right), \tag{9}$$

where

$$a_{ij}(\mathbf{q}) = \begin{cases} \dfrac{\rho_h\left\|\mathbf{q}_j - \mathbf{q}_i\right\|_\sigma}{r_\alpha} & j \neq i, \\ 0 & j = i. \end{cases} \tag{10}$$

And $a_{ij}(\mathbf{q})$ forms the spatial adjacency matrix.

The description of $\mathbf{f}_i^d$ is shown as follows:

$$\mathbf{f}_i^d = c_1(\mathbf{q}_l - \mathbf{q}_i) + c_2(\mathbf{p}_l - \mathbf{p}_i), \tag{11}$$

where c_1 and c_2 are the coefficient of the virtual leader, and only the position matching or speed matching between the cluster members and the virtual leader can be considered in the assignment process to make $c_1 = 0$ or $c_2 = 0$.

2.2. Virtual Leader Model with Direction Noise. In this paper, during the motion of the cluster to the destination, the position of the virtual leader at each time is taken as the center of the cluster. So, the position matching parameter $c_1 = 0$. Considering only the speed matching with the virtual leader, the speed magnitude $\|\mathbf{p}_l(t)\|$ and direction vector $\mathbf{n}_l(t)$ of the virtual leader at time t are given as follows:

$$\|\mathbf{p}_l(t)\| = \gamma(t)\frac{\|\mathbf{q}_l(t) - \mathbf{q}_d(t)\|}{T_t - t}, \tag{12}$$

$$\mathbf{n}_l(t) = [\cos(\theta(t)), \ \sin(\theta(t))]^T, \tag{13}$$

where T_t represents the total time for the cluster to move to the destination. $\mathbf{q}_l(t)$ represents the position of the virtual leader at time t, that is, the position of the cluster center. $\mathbf{q}_d(t)$ indicates the destination location at time t. $\gamma(t)$ is the speed compensation parameter, which meets $\gamma(t) \geq 1$. $\theta(t)$ is the motion direction of the virtual leader at time t. Because there is noise when obtaining $\theta(t)$; it is a random variable.

Generally, as the cluster approaches the destination, the more accurately the cluster detects the position information of the destination, the smaller the noise of direction would be. Therefore, we model $\theta(t)$ as a Gaussian distribution that obeys time-varying variance, that is, $\theta(t) \sim N(\bar{\theta}(t), R_\theta(t))$. Where $\bar{\theta}(t)$ is the accurate direction and $R_\theta(t)$ is its time-varying variance, which is defined as follows:

$$R_\theta(t) = \delta\|\mathbf{q}_l(t) - \mathbf{q}_d(t)\|^2, \tag{14}$$

where δ is the noise variance control coefficient.

Due to the influence of noise, the speed of the virtual leader moving to the destination is always a component of $\mathbf{p}_l(t)$, so it is necessary to add the coefficient $\gamma(t)$ to compensate. The $\gamma(t)$ is related to $R_\theta(t)$, and $\gamma(t) = f_\gamma(\mathbf{q}_l(t))$. The mean value of noise is compensated, so the following results can be obtained by solving $f_\gamma(\mathbf{q}_l(t))$.

$$\begin{aligned} f_\gamma(\mathbf{q}_l(t)) &= 1/\cos\left(\int_{\bar{\theta}(t)}^{\bar{\theta}(t)+\pi}\frac{\theta(t)-\bar{\theta}(t)}{\sqrt{2\pi R_\theta(t)}}e^{-\left(\theta-\bar{\theta}(t)\right)^2/2R_\theta(t)}d\theta(t)\right) \\ &= 1/\cos\left(\left(1-e^{-\pi^2/2R_\theta(t)}\right)\sqrt{\frac{R_\theta(t)}{2\pi}}\right). \end{aligned} \tag{15}$$

Generally, the value of δ is very small to guarantee $\|\mathbf{q}_l(t) - \mathbf{q}_d(t)\|\sqrt{\delta} < \sqrt{\pi}$. When this is true, $\gamma(t) \geq 1$. According to Equation (15), as the virtual leader approaches the destination, $R_\theta(t)$ approaches 0 and $\gamma(t)$ approaches 1.

3. Cluster Motion Intention Recognition Method

3.1. Intention Recognition Process. Because splitting or merging behavior would occur in the process of cluster moving to the destination, we divide the cluster target motion intention recognition into two parts: destination prediction and cluster identification. We would use the deep echo state network optimized by adaptive genetic algorithm (AGADESN) and DBSCAN clustering algorithm to solve the above two problems. The method process is shown in Figure 1.

As shown in Figure 1, we use the improved Olfati-Saber model with direction noise to generate a cluster motion trajectory dataset, which is divided into two parts: training dataset and test dataset. The datasets are input into AGADESN. The training dataset is used for network training and parameter optimization, and the test dataset is used for destination prediction. The destination prediction output and the predicted destination distance are generated through this process. The predicted destination distance, position distance, and speed distance of the test set are fused to produce the motion similarity distance (MSD). Based on the MSD, the DBSCAN algorithm is used to cluster the targets, and the cluster identification output is generated. The specific operation steps of intention recognition method would be described in the next section.

3.2. Destination Prediction Based on AGADESN

3.2.1. Method of Destination Prediction by DESN. Because the target trajectory is constantly updated to the destination in the process of target motion, we need to obtain multiple destination predictions at different stages of the trajectory. To this end, we design a 1-dimensional sliding window structure, as shown in Figure 2. The sliding window length is l_w, and each sliding length is l_s. The first prediction starts when the obtained trajectory length meets the sliding window length. When the update length of track points meets the sliding length, the sliding window updates the training and test data for the second prediction. If the target trajectory update ends after the mth prediction, all destination predictions are completed on the whole trajectory.

The DESN structure used for each prediction is shown in Figure 3. DESN combines echo state network (ESN) with deep learning idea, and adds multiple reservoir structures as hidden layers on the basis of ESN, so that DESN can map more complex time series characteristics.

The working mechanism of DESN is as follows. During the mth prediction, the external data enters the input layer and enters reservoir 1 after being weighted by the input weight $\mathbf{W}_{\text{in}}^{(1)}$. The state of reservoir 1 at the previous time is weighted by the weight $\mathbf{W}^{(1)}$ and summed with the received weighted external input. After the activation function, a new state is generated as the state $\mathbf{x}^{(1)}(n)$ of reservoir 1 at the current time, and the state is used as the input of reservoir 2.

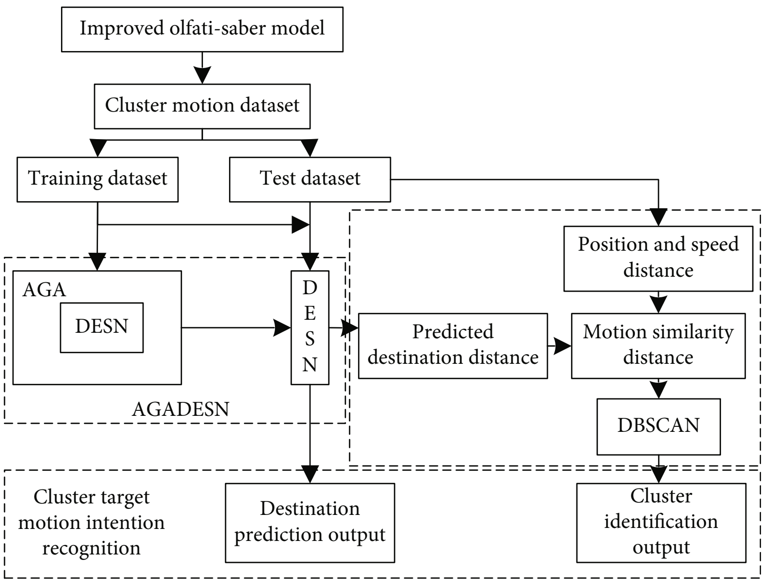
Improved olfati-saber model
Cluster motion dataset
Training dataset
Test dataset
AGA
DESN
D E S N
AGADESN
Position and speed distance
Predicted destination distance
Motion similarity distance
DBSCAN
Cluster target motion intention recognition
Destination prediction output
Cluster identification output

FIGURE 1: Intention recognition method process.

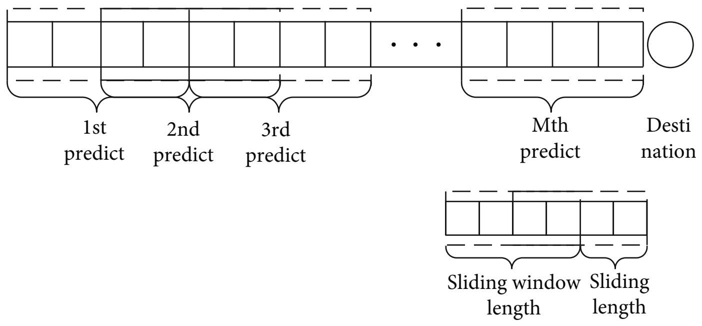

FIGURE 2: 1-dimensional sliding window structure.

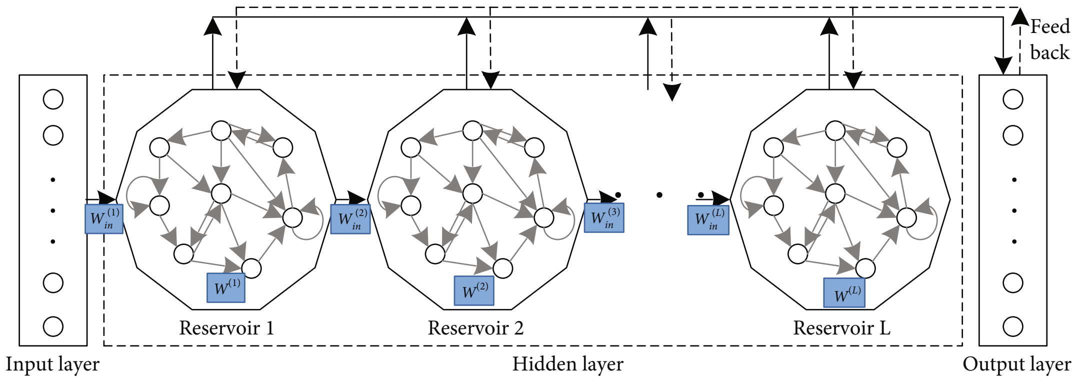

FIGURE 3: Deep echo state network structure.

After repeating the above work process, the state $\mathbf{x}^{(L)}(n)$ of the last reservoir is obtained. The set of all reservoir states is recorded as $\mathbf{x}(n) = [(\mathbf{x}^{(1)}(n))^T, (\mathbf{x}^{(2)}(n))^T, \cdots (\mathbf{x}^{(L)}(n))^T]^T$, and the output is generated by the activation function after the output layer is affected by the output weight. Among them, the maximum number of layers L is called depth, and the network requires a total of $N_n = N_{n,1} + N_{n,2} + \cdots + N_{n,L}$ neurons.

Equations ((16)–(19)) are the mathematical model of the DESN.

$$\boldsymbol{x}_{\text{in}}^{(l)}(n) = \boldsymbol{W}_{\text{in}}^{(l)} \boldsymbol{i}^{(l)}(n) + \boldsymbol{W}^{(l)} x^{(l)}(n-1) + \boldsymbol{b}^{(l)}, \tag{16}$$

$$\mathbf{i}^{(l)}(n) = \begin{cases} \mathbf{u}(n), l = 1, \\ \mathbf{x}^{(l-1)}(n), l > 1, \end{cases} \tag{17}$$

$$\mathbf{x}^{(l)}(n) = \left(1 - a^{(l)}\right)\mathbf{x}^{(l)}(n-1) + a^{(l)} f^{(l)}\left(\mathbf{x}_{\text{in}}^{(l)}(n)\right), \tag{18}$$

$$\mathbf{y}(n) = g\left(\mathbf{W}^{\text{out}}\mathbf{x}(n)\right). \tag{19}$$

In Equation (16), taking the reservoir l as an example, $\mathbf{W}_{\text{in}}^{(l)}$ and $\mathbf{W}^{(l)}$ represent the input weight and reservoir weight, respectively. $\mathbf{b}^{(l)}$ represents the noise of the reservoir, and $\mathbf{i}^{(l)}(n)$ is the input data. $\mathbf{x}^{(l)}(n)$ is the updated reservoir state, and $\mathbf{x}_{\text{in}}^{(l)}(n)$ is the weighted input data of the reservoir. In Equation (17), $\mathbf{u}(n)$ represents the external input data. In Equation (18), $a^{(l)}$ represents the leakage parameter of the reservoir l, and $a^{(l)} \in (0, 1]$. $f^{(l)}(\cdot)$ represents the activation function. Generally, $f^{(l)}(\cdot)$ is selected as the hyperbolic tangent function. In Equation (19), $\mathbf{y}(n)$ represents the output data, and $\mathbf{W}^{\text{out}}$ is the output weight. $g(\cdot)$ represents the activation function of the output layer, and $g(x) = x$ is usually selected.

DESN's training and test process are as follows. The training process of the mth prediction is essentially the training of the output weight matrix $\mathbf{W}^{\text{out}}$. Firstly, the network parameters are initialized, and the random algorithm is used to generate the input weight $\mathbf{W}_{\text{in}}^{(l)}$, reservoir weight $\mathbf{W}^{(l)}$, and noise $\mathbf{b}^{(l)}$ of the DESN, which remain unchanged during the training process. Secondly, if the trajectory sequence is $\mathbf{Q}^m = \{\mathbf{Q}_1^m, \mathbf{Q}_2^m, \cdots, \mathbf{Q}_N^m\}$ of N targets in the sliding window at the mth prediction, the trajectory sequence of the target i is as follows:

$$\mathbf{Q}_i^m = \left(\mathbf{q}_i((m-1)\cdot l_w + 1), \mathbf{q}_i((m-1)\cdot l_w + 2), \cdots, \mathbf{q}_i(m\cdot l_w)\right). \tag{20}$$

Divide $\mathbf{Q}^m$ into training sample $\mathbf{Q}_{tr}^m$ and test sample $\mathbf{Q}_{te}^m$. Input the training sample $\mathbf{Q}_1^m, \mathbf{Q}_2^m, \cdots, \mathbf{Q}_{S_{tr}}^m$ one by one, so as to obtain all reservoir state matrices $\mathbf{X} = [\mathbf{x}(1), \mathbf{x}(2), \cdots, \mathbf{x}(S_{tr})]$. Finally, calculate the training output weight $\mathbf{W}^{\text{out}}$. If the destination prediction position output corresponding to the training sample is $\mathbf{Y} = [\mathbf{y}(1), \mathbf{y}(2), \cdots, \mathbf{y}(S_{tr})]$, the output weight of the DESN for destination prediction can be obtained according to Equation (21).

$$\mathbf{W}^{\text{out}} = \mathbf{Y}\mathbf{X}^T\left(\mathbf{X}\mathbf{X}^T + \alpha\mathbf{E}\right)^{-1}, \tag{21}$$

where α represents a small positive number [38], and $\mathbf{E}$ represents the identity matrix. Among them, the destination prediction of the target i is as follows:

$$\mathbf{y}(i) = \left(\mathbf{q}_{d,i}((m-1)\cdot l_w + 1), \mathbf{q}_{d,i}((m-1)\cdot l_w + 2), \cdots, \mathbf{q}_{d,i}(m\cdot l_w)\right). \tag{22}$$

$\mathbf{q}_{d,i}(t)$ represents the predicted destination position of the ith trajectory sample at time t. After the training, input the test sample $\mathbf{Q}_{te}^m$ to test the learning effect of the DESN, which can output the destination prediction according to Equation (19) under the trained output weight $\mathbf{W}^{\text{out}}$.

3.2.2. Optimization of DESN Parameters by Adaptive Genetic Algorithm. DESN has reliable time series prediction ability. However, like other deep learning algorithms, its prediction effect is limited by the selection of parameters. The parameters that have the greatest impact on the effect of the algorithm include the number of reservoirs N_l, the number of reservoir nodes N_r, and the spectral radius ρ. ρ is one of the most important central parameters representing the reservoir weight matrix.

In order to solve the problem of choosing DESN parameters, we use an adaptive genetic algorithm (AGA) with variable mutation rate to optimize the parameters. Compared with traditional genetic algorithm, it can avoid premature convergence and fall into local optimization. The mutation rate of the AGA uses pseudoderivative to consider the time that GA stays at a certain point. The longer the algorithm stays at the local optimal value, the greater the possibility of mutation.

Taking the mth prediction as an example, firstly, the parameters of the DESN model are chromosome encoded to correspond to the genetic operator of the AGA. The encoded chromosome of DESN model parameters is as follows:

$$P = [N_l, N_r, \rho]. \tag{23}$$

Secondly, the fitness function is established to guide the search process of AGA. The design of fitness function affects the convergence speed and prediction accuracy of the algorithm. Generally, the fitness function is converted from the error function of DESN model. For the DESN model, the destination prediction error is naturally regarded as the fitness function. Therefore, the corresponding fitness function of the mth prediction is as follows:

$$F(P) = \frac{1}{S_{tr}\times l_w}\sum_{i=1}^{S_{tr}}\sum_{j=1}^{l_w}\left\|\mathbf{q}_{d,i}((m-1)\cdot l_w + j) - \hat{\mathbf{q}}_{d,i}((m-1)\cdot l_w + j)\right\|^2, \tag{24}$$

where $F(P)$ represents the average prediction error of destination position when chromosome P is the DESN model parameter.

Finally, according to the chromosome and fitness function established by AGADESN model, the best chromosome is selected through adaptive genetic operation. In the model, the selection operation [39] and crossover operation [40] are consistent with the traditional GA. In the mutation operation, the adaptive mutation rate based on the derivative of fitness function to generation is as follows:

$$r_m = 2\times(\text{sigmoid}(g) - 0.5), \tag{25}$$

where r_m is the adaptive mutation rate, and sigmoid function is defined as follows:

$$\text{sigmoid}(g) = \frac{1}{1 + e^{-g}}, \tag{26}$$

where g is defined as follows:

$$g = g_c - g_o. \tag{27}$$

Here, g_c represents the current generation, and g_o represents the oldest generation with the same optimal fitness. g, thus, is inversely related to the derivative of the fitness function with respect to generation.

The implementation steps of AGADESN are shown in Figure 4.

As the figure shows, the specific steps are as follows:

Step 1. Input training data, initialize DESN model parameters N_l, $N_{r,}$ and ρ.

Step 2. Encode the parameters of DESN model and initialize the chromosomes in the population.

Step 3. The best chromosome is selected by genetic manipulation and fitness evaluation.

Step 4. Decode the optimal chromosome to obtain the optimal parameters of the DESN model.

Step 5. Input training data and train DESN model weight $\mathbf{W}^{\text{out}}$.

Step 6. Input test data to obtain destination prediction results.

3.3. Cluster Identification Based on DBSCAN

3.3.1. Definition of Motion Similarity Distance. According to the cluster collaborative motion model, in the process of moving towards the destination, the members of the same cluster are close to each other and have similar speed vectors, which are the basis for us to cluster the target to identify the cluster. However, a fact that may be ignored is that the same cluster members also have similar destination position predictions, which provide additional information for us to identify clusters.

In 2-dimensional case, the motion state vector of target i at time t is $\mathbf{S}_{t,i}(t) = [q_{i,x}(t), p_{i,x}(t), q_{i,y}(t), p_{i,y}(t)]^T$, and the prediction value of destination is $\mathbf{q}_{d,i}(t) = [q_{d,i,x}(t), q_{d,i,y}(t)]^T$. Due to the sliding window structure, there may be more than one predicted value of the track point at time t, so $\mathbf{q}_{d,i}(t)$ is the mean value of all predicted values, as shown in Equation (28).

$$\mathbf{q}_{d,i}(t) = \frac{1}{Z}\sum_{j=1}^{Z} \mathbf{q}_{d,i}^{(j)}(t), \tag{28}$$

where Z is the number of destination position predicted values at the track point at time t. $\mathbf{q}_{d,i}^{(j)}(t)$ represents the jth predicted value.

We define a new distance measure, which is called motion similarity distance (MSD). The MSD between member i and member j is shown in Equation (29).

$$\begin{aligned} &f\left(\left[\mathbf{S}_{t,i}(t), \mathbf{q}_{d,i}(t)\right], \left[\mathbf{S}_{t,j}(t), \mathbf{q}_{d,j}(t)\right]\right) \\ &\quad = 1 - e^{-\left(w_1 R_v^2/2\sigma_{R_v}^2 + w_2 R_d^2/2\sigma_{R_d}^2 + w_3 R_p^2/2\sigma_{R_p}^2\right)}. \end{aligned} \tag{29}$$

Among them, R_v, $R_{d,}$ and R_p are the speed distance, position distance, and predicted destination distance, respectively. The weight $w_i \geq 0 \quad (i = 1, 2, 3)$ represents the importance of different distances, and $\sum_i w_i = 1$. σ_{R_v}, σ_{R_d}, and σ_{R_p} are the standard deviation of the corresponding distance. The distances are calculated as follows:

$$\begin{aligned} R_v &= \left(\left(p_{i,x}(t) - p_{j,x}(t)\right)^2 + \left(p_{i,y}(t) - p_{i,y}(t)\right)^2\right)^{1/2}, \\ R_d &= \max\left(0, \left(\left(q_{i,x}(t) - q_{j,x}(t)\right)^2 + \left(q_{i,y}(t) - q_{j,y}(t)\right)^2\right)^{1/2} - D_{e1}\right), \\ R_p &= \max\left(0, \left(\left(q_{d,i,x}(t) - q_{d,j,x}(t)\right)^2 + \left(q_{d,i,y}(t) - q_{d,j,y}(t)\right)^2\right)^{1/2} - D_{e2}\right), \end{aligned} \tag{30}$$

where D_{e1} describes the expected distance of members. D_{e2} describes the extended range of destinations.

3.3.2. Principle of DBSCAN Cluster Identification. DBSCAN is an effective density-based clustering algorithm, which can describe the density of data distribution, and then identify clusters of arbitrary shape and effectively distinguish noise points. DBSCAN algorithm sets a field with a radius of *Eps* around each target data point. If the number of other target data points in a target field reaches the set threshold *MinPts*, it is considered that the target belongs to the cluster and is the core point. If there are no other target data points in the neighborhood of a target, the target is considered to be a noise point. If there are target data points in the target field, but the number of data points does not meet *MinPts*, it is considered that the target is a boundary point. Figure 5 shows the schematic diagram of the DBSCAN algorithm.

As shown in Figure 5, the radius of the circle is the clustering radius *Eps* and set MinPts = 2. It can be seen from the figure that the number of data points in the neighborhood of targets (2, 3, 4, 5, and 6) is greater than MinPts, so they are core points. There are only points 2 in the neighborhood of data point 1, so point 1 is a boundary point. There are no other targets in the neighborhood of data point 7, so it is a noise point. We can obtain that (1, 2, 3, 4, 5, and 6) belong to the same cluster. The DBSCAN algorithm flow is as follows:

4. Experiments and Results

4.1. Generate Datasets. In order to verify the effectiveness of the algorithm proposed in this paper, we generate a cluster cooperative motion simulation dataset based on the

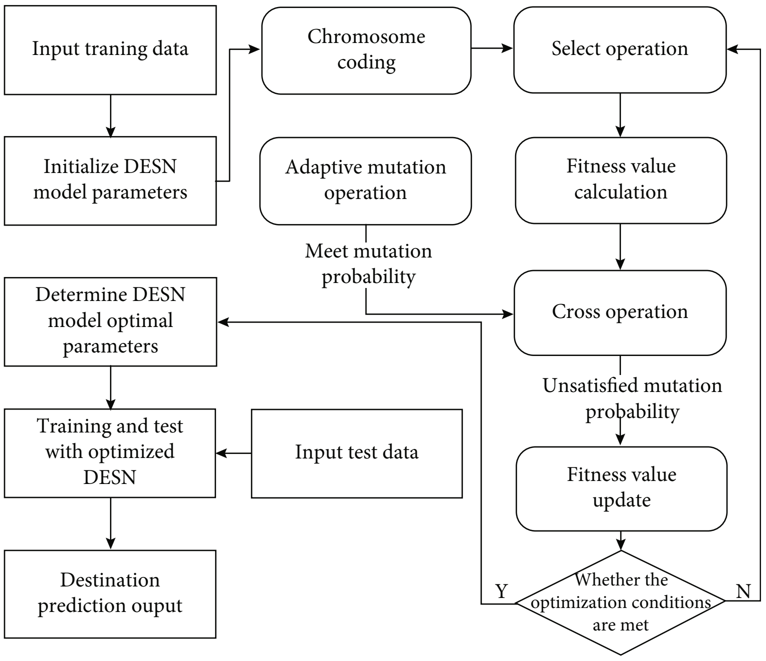
Input traning data
Chromosome coding
Select operation
Initialize DESN model parameters
Adaptive mutation operation
Fitness value calculation
Meet mutation probability
Determine DESN model optimal parameters
Cross operation
Unsatisfied mutation probability
Training and test with optimized DESN
Input test data
Fitness value update
Destination prediction ouput
Y
Whether the optimization conditions are met
N

Figure 4: Process for implementing AGADESN algorithm.

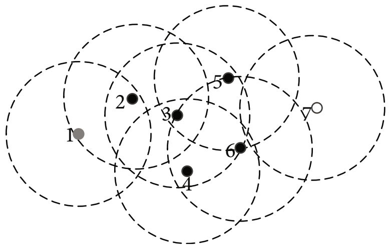

Figure 5: Schematic diagram of the DBSCAN algorithm.

improved Olfati-Saber model. This algorithm uses the combination of AGADESN and DBSCAN algorithms to identify the motion intention of cluster targets. The total number of targets in the field of view is $N = 15$.Targets are released in the range of $[0, 20] \times [0, 20]$(m2), and the initial speed in both x and y directions is 0 m/s. The number of clusters is random, but there are at least 3 targets in a cluster, that is, the number of clusters in the field of view is $N_c \leq 5$. Total movement time $T_t = 100$s, and sampling interval $\tau = 0.1$s. 5 destinations are set to form a destination set, and the position of each destination is shown in Table 1.

In the simulation, the destination of the cluster motion is randomly determined and can be changed during the cluster motion, but in order to ensure the stability of the trajectory, the destination can be changed again only after the destination is determined for the first time or changed for at least 25 s.

Set the Olfati-Saber model parameters $a = 1$ and $b = 5$, then the corresponding $c = 2\sqrt{5}/5$. Set parameter $\varepsilon = 0.85$, select $d = 15$ and $r = 95$, then the corresponding $d_\alpha = 15.1$ and $r_\alpha = 101.9$. Only consider the speed collaboration with the virtual leader, and set the parameters $h = 0.2$, $c_1 = 0$, and $c_2 = 0.2$. Set the noise variance control coefficient $\delta = 5 \times 10^{-4}$.

Under the above simulation conditions, 100 cluster motion scenes are generated, and a total of 1500 tracks are obtained; 80% of which are used for training and 20% for test. The generated dataset is shown in Figure 6.

A cluster movement scene is taken out from the training set, as shown in Figure 7, and the cluster motion speed is shown in Figure 8.

As can be seen from Figures 7 and 8, the motion of each target is chaotic at the initial stage of release. With the mutual cooperation of the targets in the same cluster, the speed of the targets in the same cluster gradually tends to be the same, and three different clusters are gradually formed. Around 33 s, the cluster moving towards destination D_1 splits and gradually forms cluster C_2 and C_3. After C_2 changes the destination and moves for a period of time, it changes the destination to D_1 again and arrives. C_3 finally arrives at D_3. At about 66 s, the clusters moving towards destination D_4 splits and gradually forms cluster C_4 and C_5. C_4 and C_5 finally arrive at D_4 and D_5, respectively. In the late stage of target motion, the speed of each target in the same cluster tends to be consistent, and the trajectory is gradually parallel, and the cluster forms a stable structure. It can also be found from Figure 8 that due to the time-varying variance of the direction noise, the mean value of the noise at the initial time is large, and the trajectory has great instability. As the target approaches the destination, the mean value of the noise approaches 0, and the cluster speed is smoother.

4.2. Destination Prediction Results. In order to verify the superiority of the proposed algorithm, we use the algorithm

```
Input: Uncertain data set $D$, Clustering radius $Eps$, Minimal number of neighboring points $MinPts$
Output: A set of clusters, types of all the points in $D$
Main function of the algorithm:
    DBSCAN($D$, $Eps$, $MinPts$)
        ClusterNum=0
    for each target point $P$ in $D$
        if $P$ is visited
        continue to next point
        end if
    mark $P$ as visited
        $Eps$-$Neighborhood$=regionQuery($P$, $Eps$)
        if sizeof($Eps$-$Neighborhood$)<$MinPts$
          mark $P$ as noise
        else
        $ClusterNum$=next cluster
        expandCluster($P$, $Eps$-$Neighborhood$, $ClusterNum$, $Eps$, $MinPts$)
        end if
      end for
    END
    regionQuery($P$, $Eps$)
          return $Eps$-$Neighborhood(P)=\{Q \in S | D(P,Q) \leq Eps\}$
        expandCluster($P$, $Eps$-$Neighborhood$, $ClusterNum$, $Eps$, $MinPts$)
        add $P$ to cluster $ClusterNum$
          for each point $P'$ in $Eps$-$Neighborhood$
            if $P'$ is not visited
              mark $P'$ as visited
              $Eps - Neighborhood'$ =regionQuery($P'$, $Eps$)
              if sizeof($Eps - Neighborhood'$)>=$MinPts$
                $Eps$-$Neighborhood$=$Eps$-$Neighborhood$ joined with $Eps - Neighborhood'$
              end if
            end if
            if $P'$ is not yet member of any cluster
              add $P'$ to cluster $ClusterNum$
            end if
end for
```

Algorithm 1: **DBSCAN algorithm flow.**

Table 1: Destination position.

Destination	Position (x, y)/m
D_1	(1400, 2040)
D_2	(1960, 2500)
D_3	(2560, 3360)
D_4	(2880, 2940)
D_5	(3440, 2680)

in this paper, the DESN algorithm optimized by genetic algorithm (GADESN) and the DESN algorithm to predict the destination of cluster targets, respectively. Set the sliding window length $l_w = 100$, and each sliding length $l_w = 20$, then $M = 46$ times of destination prediction is required to make the sliding window traverse the whole trajectory. The calculation method of M is as follows:

$$M = 1 + \frac{(T_t/\tau - l_w)}{l_s}. \quad (31)$$

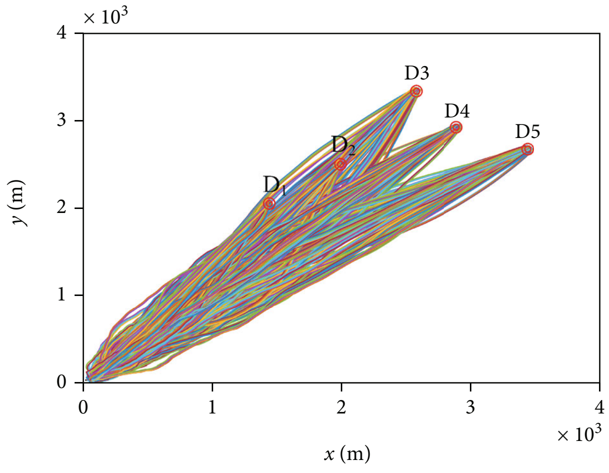

Figure 6: Cluster motion dataset.

The root mean square error (RMSE) is used to evaluate the prediction performance of the AGADESN model, which is defined as follows:

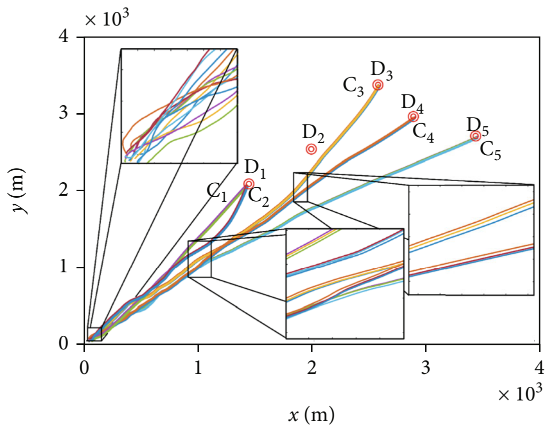

Figure 7: A cluster motion scenario.

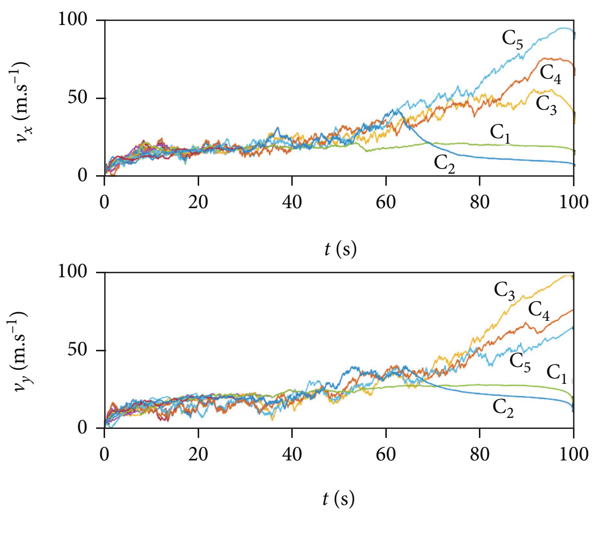

Figure 8: Cluster motion speed of the above scenario.

$$\mathrm{RMSE}(t) = \sqrt{\frac{1}{Z}\sum_{i=1}^{Z}\left\|\mathbf{q}_{d,i}^{(i)}(t) - \hat{\mathbf{q}}_{d,i}(t)\right\|^2}. \quad (32)$$

Under the above sliding window setting, $Z \in \{1, 2, 3, 4, 5\}$.

Set reservoir leakage parameters $a^{(l)} = 0.9$. The number of reservoirs N_l, the number of neurons N_r in each reservoir, and the spectral radius ρ are given by AGA in AGADESN algorithm and by GA in GADESN algorithm. In the DESN algorithm, we set $N_l = 5$, $N_r = 200$, and $\rho = 0.8$. Taking the 46th prediction as an example, the iteration trajectory of the AGA and the GA are shown in the following figure.

It can be seen from Figure 9 that the fitness value of the adaptive genetic algorithm has converged when iterating to generation 46, with a fitness value of 42.6, and the genetic algorithm has converged when iterating to generation 74, with a fitness value of 48.3. Adaptive genetic algorithm has faster convergence speed and better fitness value, so AGADESN model has faster iteration speed and better prediction accuracy.

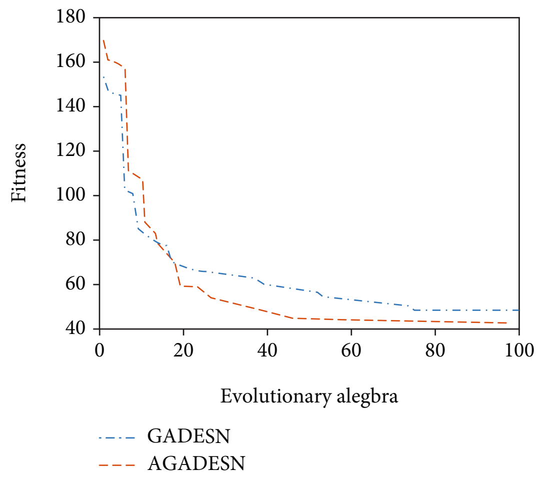

Figure 9: Adaptive genetic algorithm and genetic algorithm fitness value iteration trajectory.

The destination position prediction RMSE of the cluster in Figure 7 is shown in the following figure.

It can be seen from Figure 10 that as the cluster approaches the destination, although the destination position prediction RMSE of AGADESN algorithm fluctuates, on the whole, RMSE tends to decrease gradually. The RMSE of all clusters can be reduced to less than 100 m before the end of the motion. Since there is no destination change in C_4, the algorithm can determine the destination very early and maintain RMSE at a low level. When the destination of C_2 changes again in the late stage of motion, the algorithm can also quickly catch the new destination, and the RMSE gradually decreases.

In the figure, the circle with the destination as the center and 300 m as the radius can distinguish different destinations in the destination set, which is called the destination range. Generally, if the destination set is known, we do not need to get the accurate destination prediction position of the target, but only need to specify which destination range the predicted value is in to determine the destination. If the mean value of all the destination position predicted values on a certain trajectory point is within the range of the real destination, we call the prediction on this trajectory point the correct prediction. Under the above definition, the condition for correct prediction in this experiment is that the mean absolute error (MAE) of all predicted values at a certain trajectory point is no more than 300 m. The MAE(t) is defined as follows:

$$\mathrm{MAE}(t) = \frac{1}{Z}\sum_{i=1}^{Z}\left\|\mathbf{q}_{d,i}^{(i)}(t) - \hat{\mathbf{q}}_{d,i}(t)\right\|. \quad (33)$$

Figure 11 gives the average RMSE of destination position prediction for all cluster in the test set by the AGADESN, GADESN, and DESN algorithms.

RMSE is computed differently from MAE and according to holder inequality, there always holds RMSE(t) ≥ MAE(t). Therefore, the RMSE threshold for correct prediction is always

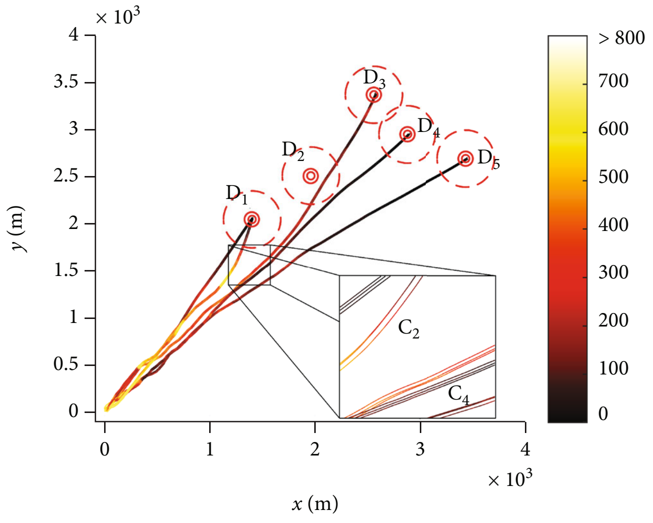

Figure 10: Destination position prediction RMSE of the cluster in Figure 7.

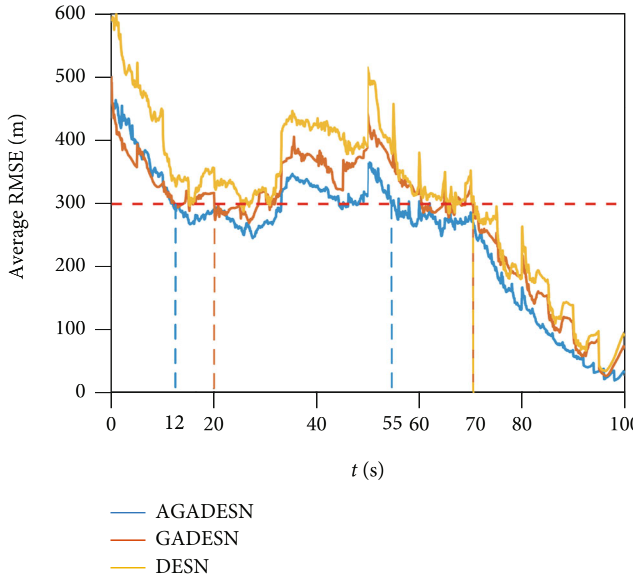

Figure 11: Average RMSE of destination position prediction for all cluster in the test set.

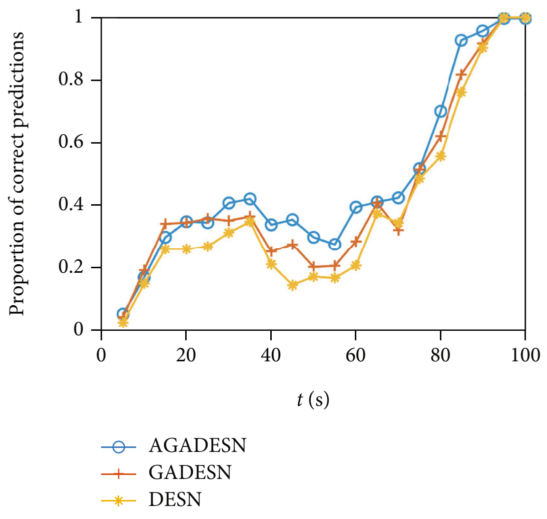

Figure 12: Proportion of correct predictions.

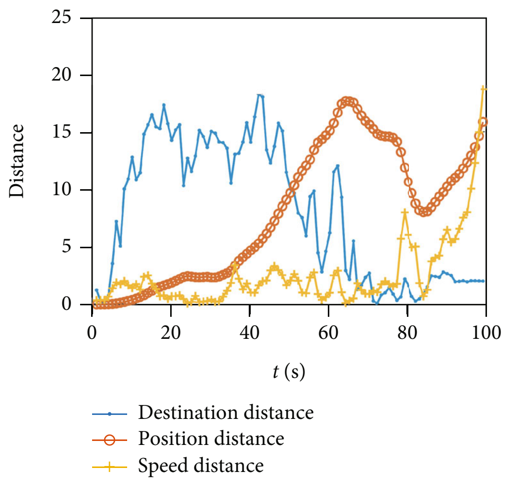

Figure 13: R_v, R_d and R_p distances of the two targets at each moment.

not less than the destination range. But without knowing how optimistic the predicted value is, we take the RMSE threshold of the correct prediction to be its lower bound 300 m, as indicated by the red dotted line in the figure.

Figure 11 shows that in the initial stage, before the target destination changes, all 3 algorithms can predict the approximate position of the destination. Comparatively, AGADESN algorithm achieves the correct prediction at 12 s, which is 8 s earlier than GADESN algorithm. Some of the targets change their destinations around 30 s and 50 s; the RMSE of 3 algorithms increases, and after recognizing the new destinations, the RMSE returns to a lower level. After a second destination change maneuver of targets occurs, the AGADESN algorithm achieves more stably correct predictions in 55 s, whereas the GADESN and DESN algorithms need 70 s. The AGADESN algorithm has a better effect, both in terms of prediction accuracy and in the time to achieve the correct prediction.

In the whole process of target motion, the correct prediction proportion of destinations of all clusters in the test set is shown in Figure 12.

In Figure 12, we plot the proportion of correct predictions of destinations for the test set clusters at 5 s intervals. It is known from the figure that the AGADESN algorithm can correctly predict about 40% of the cases when no destination change occurs. When the destination of the cluster motion is changed, the proportion of correct predictions decreases for 3 algorithms. However, the proportion of correct predictions improves faster for the AGADESN algorithm after 55 s. Throughout the trajectory, the proportion of correct predictions by the AGADESN algorithm proposed in this paper is generally higher than that by the other comparison algorithms, increasing by 8.39% the proportion of correct predictions by the DESN algorithm.

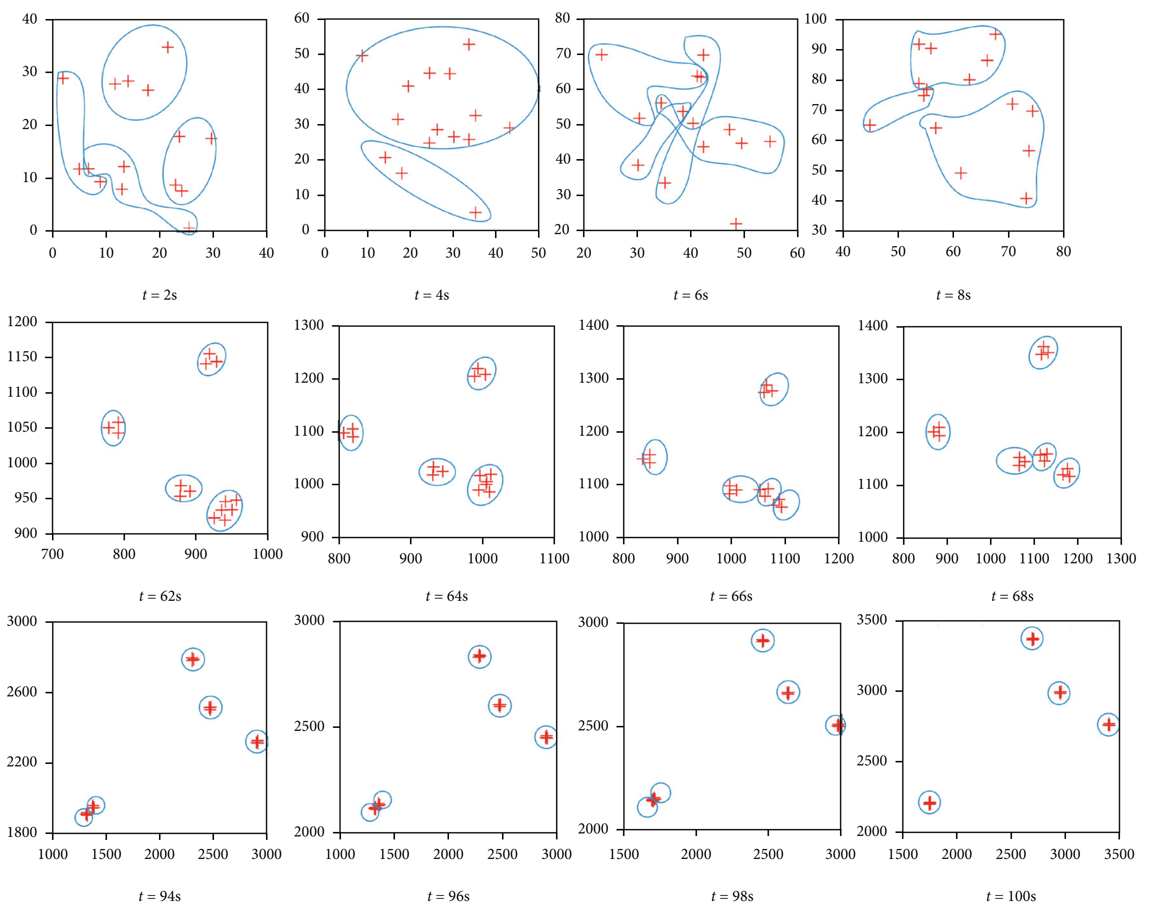

FIGURE 14: Cluster identification results at specific time.

4.3. Cluster Identification Results. To demonstrate the effectiveness of DBSCAN algorithm based on the motion similarity distance proposed in this paper for cluster identification, we set the distance parameters $\sigma_{R_v} = 0.7$, $\sigma_{R_d} = 100$, $\sigma_{R_p} = 600$, $D_{e1} = 15$, and $D_{e2} = 300$ after obtaining the destination position predictions. Solving for the R_v, R_d, and R_p distances for each moment of target 1 and target 7 belonging to different clusters are shown in Figure 13. Among them, target 1 belongs to C_3; initial destination is D_1, and final destination is D_3 after 33 s. Target 7 belongs to C_4; initial destination is D_5, and final destination is D_4 after 32 s.

It is known from Figure 13 that the initial moment, because the target release point is concentrated, the two targets are relatively close together, and the speed difference is not great, so that the R_v and R_d of the two targets are both smaller in the motion early stage. But because the predicted destinations of the two targets are quite different, the R_p distance difference is obvious in the motion early stage. In the motion late stage, after the two targets turning, the R_p distance starts to decrease due to the closer distances between the D_3 and D_4. But R_v and R_d distances increase. Thus, the MSD defined in this paper, which fuses the R_v, R_d, and R_p distances, can effectively compensate for the main time of action of the three distances, such that targets belonging to different clusters have large distances consistently throughout the motion.

Set DBSCAN algorithm parameters Eps = 0.3; MinPts = 2, and $w_1 = w_2 = w_3 = 0.33$. Clusters are identified throughout using the DBSCAN algorithm based on the MSD, and Figure 14 shows the cluster identification results for $t = 2, 4, 6, 8\,s$ in the motion early stage; $t = 62, 64, 66, 68\,s$ in the splitting stage, and $t = 94, 96, 98, 100\,s$ in the motion late stage.

In the early stage of cluster motion, the release area of cluster is small, and the target belonging to the same cluster has not completed speed collaboration. Because the MSD integrates position, speed, and predicted destination distances, the result of cluster identification has crossover in the position plane. When $t = 8\,s$, different clusters begin to differentiate in the field of view of the sensor. When cluster splits, different cluster positions are far away from each other, so the algorithm in this paper can easily distinguish different clusters and effectively identify the newly born clusters. In the late stage of cluster motion, the algorithm can maintain the accuracy of cluster identification, but when clusters C_1 and C_2 arrive at D_1 at the same time, the algorithm identifies C_1 and C_2 as the same cluster. This is because the speed distance when the cluster arrives at the destination plays a major role in distinguishing clusters,

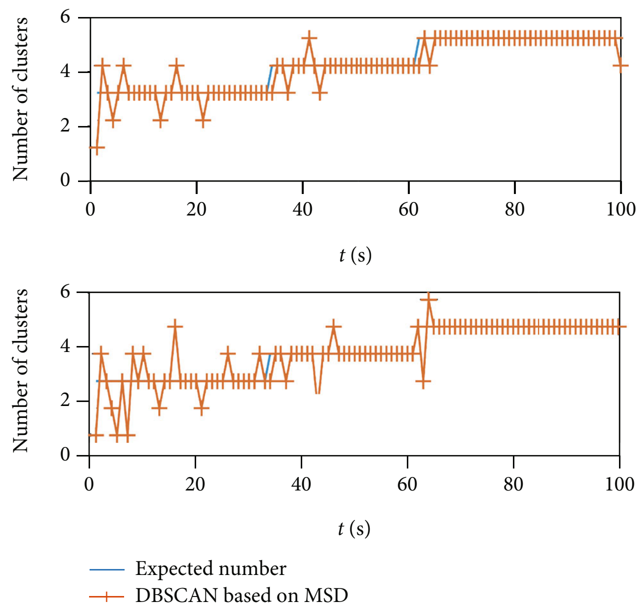

Figure 15: Comparison of cluster identification results based on the MSD and the PSD.

and the contribution of the speed distance to the MSD decreases when the predicted destination distance is introduced, which has an impact on the identification result. But on the other hand, if two clusters arrive at the same destination at the same time, whether the two clusters merge or not would not affect our further decision.

Figure 15 shows the comparison of cluster identification results by DBSCAN algorithm based on the proposed MSD and based on only position and speed fusion distance (PSD), where the DBSCAN algorithm based on the PSD is set to $w_1 = w_2 = 0.5, w_3 = 0$.

As can be seen from Figure 15, at the initial stage of cluster motion, due to the small distance difference between targets, the identification results of DBSCAN algorithm are often inaccurate. When the target is splitting, the identification result lags behind the time of splitting and fluctuates. By comparison, DBSCAN algorithm based on the MSD has higher identification accuracy and stability than that only considering position and speed distance. Especially in the initial stage of cluster motion, the introduction of the predicted destination distance significantly makes up for the deficiency of only considering position and speed distance.

4.4. Discussion

4.4.1. Selection of Eps. The selection of *Eps* has an impact on DBSCAN algorithm. In order to evaluate the significance of *Eps* influence, we increase *Eps* from 0.01 to 1 with the step size of 0.01, and set appropriate distance parameters for all cluster scenarios in the test set to conduct cluster identification experiments. The correct identification probability of cluster number is shown in Figure 16.

As can be seen from Figure 16, although *Eps* can be set arbitrarily within [0.01, 1), the DBSCAN algorithm can obtain a good correct identification probability under any

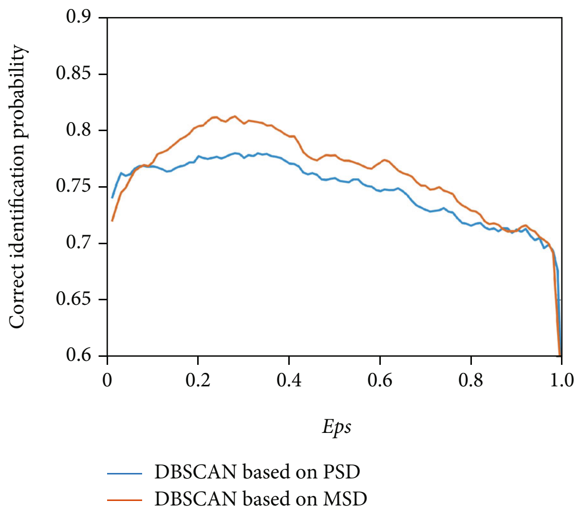

Figure 16: Correct identification probability under different *Eps*.

Eps. However, with the increase of *Eps*, the probability still shows a process of first increasing and then decreasing. When *Eps* is set in (0.3, 0.4), the correct identification probability is the highest. The analysis shows that *Eps* setting value is related to a variety of factors and is most closely related to the distance between clusters during the whole motion of clusters. When clusters are close to each other, if *Eps* is set too small, members in the same cluster may be considered as belonging to different clusters, resulting in a large number of identified clusters. On the contrary, if *Eps* is set too large, the number of identified clusters may be less. Therefore, for the whole process of cluster motion, the adaptive setting of *Eps* can achieve the best cluster identification effect.

4.4.2. Algorithm Runtime Evaluation. To evaluate the feasibility of the algorithm on large-scale data sets, Figures 17 and 18 show the running time of each intention recognition of the algorithm in the experiment.

Figure 17 shows the training time of AGADESN algorithm for each destination prediction. It can be seen from Figure 17 that the average training time for each prediction of AGADESN algorithm is 607 s, of which the iteration time for adaptive genetic algorithm is 83 s, and the time for DESN training is 524 s. It can be seen that because the adaptive genetic algorithm embeds the DESN training process, the algorithm needs to calculate the DESN training error in each iteration, so the DESN training time accounts for the main part of the AGADESN algorithm training time. This also leads to the long training time of the algorithm. The adaptive genetic algorithm proposed in the paper uses adaptive mutation rate, and its genetic operation iteration time is short. Comparing the training time of each destination prediction, it can be seen that when the target is maneuvering, the time required for algorithm training increases because the destination position is unknown after maneuvering. When the target approaches the destination, the cluster moves straight toward the destination with less noise, and the algorithm training time is reduced.

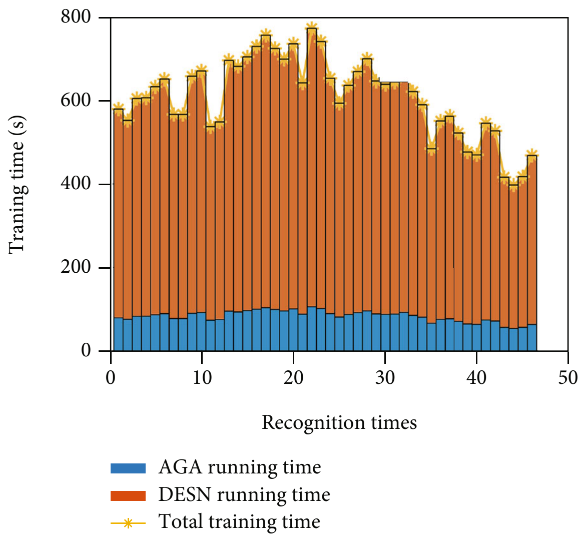

FIGURE 17: Training time of AGADESN algorithm.

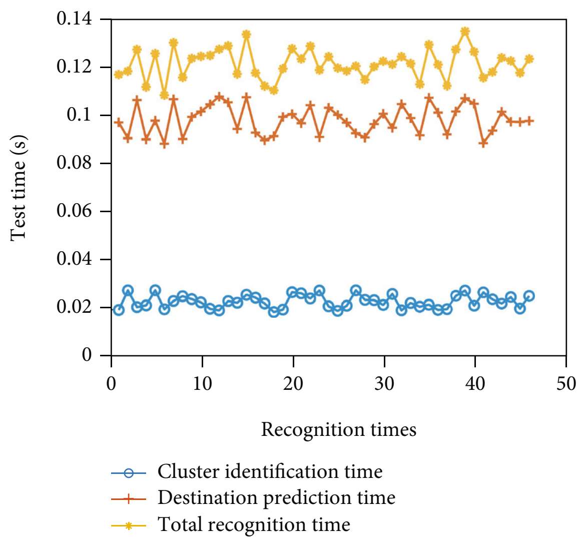

FIGURE 18: Average time of intention recognition under single test scenario.

Figure 18 shows the average recognition time of our algorithm for the cluster intention in a single test scenario. The average time of intention recognition of the trained network in a single scene is 0.1221s, including 0.0988 s for destination prediction and 0.0233 s for cluster identification. It can be seen from Figure 18 that the algorithm has a fast intention recognition speed for 15 targets in a single test scenario, and the destination prediction time accounts for the main part of the algorithm's running time. Because DBSCAN algorithm has no deep structure, it has higher efficiency. Its clustering identification speed is fast and can meet the real-time requirements.

From the above analysis, it can be seen that the trained network can quickly identify the cluster motion intention, which makes the algorithm applicable to the prediction of large-scale cluster motion intention. However, because AGADESN algorithm directly uses the training error of DESN as the fitness evaluation function in the training process, it takes a long time for a single iteration of AGADESN, which makes the algorithm unable to meet the requirements of online real-time training, which limits the training speed of the algorithm in large-scale data sets.

4.4.3. Exploration of Further Improving the Accuracy of Intention Recognition. In this paper, in order to accurately simulate more the cluster motion, we introduce the motion direction noise with time-varying variance on the basis of the basic Olfati-Saber model. However, the cluster motion trajectory noise makes the relationship between the motion characteristics and the destination more unclear, which has a negative impact on the prediction of the cluster motion destination. In order to alleviate the interference of noise and extract more motion features related to the destination, we try to decompose the cluster motion trajectory signal by using the variational mode decomposition (VMD) algorithm [41]. Due to the independence of different modality data, it is possible to further improve the accuracy of destination prediction by inputting different modality data into the circular convolution attention network to fuse different features for destination prediction [42].

We define the motion similarity distance and introduce the destination prediction distance into the position speed distance to judge whether multiple targets belong to the same cluster, which achieves better cluster recognition effect. However, when the cluster splits, the effect of DBSCAN algorithm in identifying clusters fluctuates and lags, which brings challenges to the real-time performance of the algorithm. During the experiment, we realize that the improvement of the clustering identification effect is essentially due to the introduction of additional information (destination information). Therefore, in order to improve the real-time performance of the cluster recognition algorithm, further information mining of the cluster motion trajectory is needed.

In terms of hardware, we can use infrared, photoelectric and other sensors to obtain the multidimensional information of the target, but we must propose better feature selection and fusion algorithms. Chen et al. [43] proposed two deep learning frameworks to explore and preserve the temporal and spatial information of EEG signals in a cascade or parallel manner and achieved good results in human intention recognition, which is enlightening to us. The basis of cluster identification is that targets with similar features should have a greater probability of being neighbours. We can use more complex convolutional neural networks to explore the local structure of clusters and integrate it into the selection of joint features [44]. Secondly, the cluster identification should be carried out in a more adaptive way, using an adaptive loss function to enhance the robustness to system noise, which may be caused by wrong destination prediction or by abnormal parameter values of DBSCAN algorithm.

5. Conclusions

In this paper, we propose a joint algorithm of AGADESN algorithm and DBSCAN clustering algorithm for motion intention recognition of cluster targets. It can simultaneously predict the moving destination of cluster targets and identify clusters in the field of view. The improved Olfati-Saber model is used to generate cluster motion, which takes into account the accuracy of destination information acquired by cluster targets, and introduces the motion direction noise with time-varying variance. In the intention prediction algorithm, we introduce the adaptive mutation rate to optimize genetic algorithm and use adaptive genetic algorithm to optimize DESN parameters. The optimized DESN is used for destination prediction of cluster targets, and a more accurate prediction effect is achieved. The prediction output information provides input for the innovatively designed MSD. Based on the MSD, we identify clusters within the field of view under the DBSCAN framework to judge in real-time whether target splitting behavior occurs, which is more robust than the DBSCAN algorithm considering only the position and speed distance. In future work, we may improve the clustering algorithm to reduce the required parameters, further improving the cluster identification performance.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

We declare that we have no financial and personal relationships with other people or organizations that can inappropriately influence our work.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grant No. 61703424.

References

[1] E. L. Martin, L. H. David, and L. James, *Handbook of Multi-Sensor Data Fusion: Theory and Practice*, CRC press, 2017.

[2] L. He, P. Bai, X. Liang, J. Zhang, and W. Wang, “Feedback formation control of UAV swarm with multiple implicit leaders,” *Aerospace Science and Technology*, vol. 72, pp. 327–334, 2018.

[3] Y. Qi, S. Zhou, Y. Kang, and S. Yan, “Formation control for unmanned aerial vehicles with directed and switching topologies,” *International Journal of Aerospace Engineering*, vol. 2016, Article ID 7657452, 8 pages, 2016.

[4] X. Xue, S. Huang, J. Xie, J. Ma, and N. Li, “Resolvable cluster target tracking based on the DBSCAN clustering algorithm and labeled RFS,” *IEEE Access*, vol. 9, pp. 43364–43377, 2021.

[5] F. Ge, K. Li, Y. Han, W. Xu, and Y. A. Wang, “Path planning of UAV for oilfield inspections in a three-dimensional dynamic environment with moving obstacles based on an improved pigeon-inspired optimization algorithm,” *Applied Intelligence*, vol. 50, no. 9, pp. 2800–2817, 2020.

[6] T. Huang, D. Huang, N. Qin, and Y. Li, “Path planning and control of a quadrotor UAV based on an improved APF using parallel search,” *International Journal of Aerospace Engineering*, vol. 2021, Article ID 5524841, 14 pages, 2021.

[7] J. Chen, C. Du, Y. Zhang, P. Han, and W. Wei, “A clustering-based coverage path planning method for autonomous heterogeneous UAVs,” *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–11, 2021.

[8] J. Chen, Y. Zhang, L. Wu, T. You, and X. Ning, “An adaptive clustering-based algorithm for automatic path planning of heterogeneous UAVs,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 9, pp. 16842–16853, 2022.

[9] J. Chen, F. Liang, Y. Zhang, T. You, and Y. Liu, “Coverage path planning of heterogeneous unmanned aerial vehicles based on ant colony system,” *Swarm and Evolutionary Computation*, vol. 69, article 101005, 2022.

[10] J. Chen, Y. He, Y. Zhang, P. Han, and C. Du, “Energy-aware scheduling for dependent tasks in heterogeneous multiprocessor systems,” *Journal of Systems Architecture*, vol. 129, article 102598, 2022.

[11] C. Reynolds, “Flocks, herds and schools: a distributed behavioral model,” in *SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, Anaheim, USA, August 1987.

[12] T. Vicsek, A. Czirok, E. Ben-Jacob, I. Cohen, and O. Shochet, “Novel type of phase transition in a system of self-driven particles,” *Physical Review Letters*, vol. 75, no. 6, pp. 1226–1229, 1995.

[13] I. D. Couzin, J. Krause, N. Franks, and S. A. Levin, “Effective leadership and decision-making in animal groups on the move,” *Nature*, vol. 433, no. 7025, pp. 513–516, 2005.

[14] R. Olfati-Saber, “Flocking for multi-agent dynamic systems: algorithms and theory,” *IEEE Transactions on Automatic Control*, vol. 51, no. 3, pp. 401–420, 2006.

[15] X. B. Guo, Z. Wang, and W. D. Hu, “Information fusion with Bayesian networks for target recognition,” *Acta Simulata Systematica Sinica*, vol. 17, no. 9, pp. 2713–2716, 2005.

[16] S. Hosseini and D. Ivanov, “Bayesian networks for supply chain risk, resilience and ripple effect analysis: a literature review,” *Expert Systems with Applications*, vol. 161, article 113649, 2020.

[17] S. Fan, J. Zhang, E. Blanco-Davis, Z. Yang, and X. Yan, “Maritime accident prevention strategy formulation from a human factor perspective using Bayesian networks and TOPSIS,” *Ocean Engineering*, vol. 210, article 107544, 2020.

[18] D. Qiao, Y. Liang, C. Ma et al., “Recognition and prediction of group target intention in multi-domain operations,” *Systems Engineering and Electronics*, pp. 1–12, 2021.

[19] J. Xue, J. Zhu, J. Xiao, S. Tong, and L. Huang, “Panoramic convolutional long short-term memory networks for combat intension recognition of aerial targets,” *IEEE Access*, vol. 8, pp. 183312–183323, 2020.

[20] H. Huang, Z. Zeng, D. Yao, X. Pei, and Y. Zhang, “Spatial-temporal ConvLSTM for vehicle driving intention prediction,” *Tsinghua Science and Technology*, vol. 27, no. 3, pp. 599–609, 2022.

[21] Y. Itoh and M. Adachi, “Chaotic time series prediction by combining echo-state networks and radial basis function networks,” in *2010 IEEE International Workshop on Machine Learning for Signal Processing*, pp. 238–243, Kittila, Finland, 2010.

[22] G. Claudio, M. Alessio, and P. Luca, “Design of deep echo state networks,” *Neural Networks*, vol. 108, pp. 33–47, 2018.

[23] T. Kim and B. R. King, "Time series prediction using deep echo state networks," *Neural Computing and Applications*, vol. 32, no. 23, pp. 17769–17787, 2020.

[24] Z. Song, K. Wu, and J. Shao, "Destination prediction using deep echo state network," *Neurocomputing*, vol. 406, pp. 343–353, 2020.

[25] X. Chen and H. Zhang, "Grey wolf optimization–based deep echo state network for time series prediction," *Frontiers in Energy Research*, vol. 10, 2022.

[26] Y. Lu, Y. Liao, L. Xu, Y. Liu, and Y. Liu, "Laplacian deep echo state network optimized by genetic algorithm," in *2021 IEEE International Conference on Information Communication and Software Engineering (ICICSE)*, Chengdu, China, 2021IEEE.

[27] M. S. Kanwal, A. S. Ramesh, and L. A. Huang, "Using computation to enhance diagnosis and therapy: a novel mutation operator for real-coded adaptive genetic algorithms," *F1000Research*, vol. 2, p. 139, 2013.

[28] L. Snidaro, I. Visentini, and K. Bryan, "Fusing uncertain knowledge and evidence for maritime situational awareness via Markov logic networks," *Information Fusion*, vol. 21, no. 1, pp. 159–172, 2015.

[29] M. Oxenham, S. Challa, and M. Morelande, "Fusion of disparate identity estimates for shared situation awareness in a network-centric environment," *Information Fusion*, vol. 7, no. 4, pp. 395–417, 2006.

[30] S. G. Meester and J. MacKay, "A parametric model for cluster correlated categorical data," *Biometrics*, vol. 50, no. 4, pp. 954–963, 1994.

[31] S. C. Johnson, "Hierarchical clustering schemes," *Psychometrika*, vol. 32, no. 3, pp. 241–254, 1967.

[32] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability*, pp. 281–297, Berkeley, USA, 1967.

[33] M. Ester, H. P. Kriegel, J. Sander, and X. W. Xu, "A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, Portland, OR, USA, 1996.

[34] X. Zhang, H. Liu, and X. Zhang, "Novel density-based and hierarchical density-based clustering algorithms for uncertain data," *Neural Networks*, vol. 93, pp. 240–255, 2017.

[35] A. Strandburg-Peshkin, C. R. Twomey, N. W. F. Bode et al., "Visual sensory networks and effective information transfer in animal groups," *Current Biology*, vol. 23, no. 17, pp. R709–R711, 2013.

[36] M. Ballerini, N. Cabibbo, R. Candelier et al., "Interaction ruling animal collective behavior depends on topological rather than metric distance: evidence from a field study," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 105, no. 4, pp. 1232–1237, 2008.

[37] R. Olfati-Saber and R. M. Murray, "Flocking with obstacle avoidance: cooperation with limited communication in mobile networks," in *42nd IEEE International Conference on Decision and Control (IEEE Cat. No.03CH37475)*, pp. 2022–2028, Maui, HI, USA, 2003.

[38] J. Y. Chew, D. Kurabayashi, and Y. Nakamura, "Echo state networks with Tikhonov regularization: optimization using integral gain," *Advanced Robotics*, vol. 29, no. 12, pp. 801–814, 2015.

[39] C. Guo, "Optimization of support vector machine time series forecast model based on genetic algorithm," *Chinese Journal of Scientific Instrument*, vol. 9, pp. 1080–1084, 2006.

[40] X. Yan, R. Gong, and Q. Zhang, "Application of SVM optimized by genetic algorithm in short-term wind speed prediction," *Power System Protection and Control*, vol. 44, no. 9, pp. 38–42, 2016.

[41] K. Dragomiretskiy and D. Zosso, "Variational mode decomposition," *IEEE Transactions on Signal Processing*, vol. 62, no. 3, pp. 531–544, 2014.

[42] D. Zhang, L. Yao, K. Chen, S. Wang, X. Chang, and Y. Liu, "Making sense of spatio-temporal preserving representations for EEG-based human intention recognition," *IEEE Transactions on Cybernetics*, vol. 50, no. 7, 2020.

[43] K. Chen, L. Yao, D. Zhang, X. Wang, X. Chang, and F. Nie, "A semisupervised recurrent convolutional attention model for human activity recognition," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 5, pp. 1747–1756, 2020.

[44] M. Luo, X. Chang, L. Nie, Y. Yang, A. G. Hauptmann, and Q. Zheng, "An adaptive semisupervised feature analysis for video semantic recognition," *IEEE transactions on cybernetics*, vol. 48, no. 2, pp. 648–660, 2018.