IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 4 September 2023, accepted 27 September 2023, date of publication 2 October 2023, date of current version 9 October 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3321270

APPLIED RESEARCH

Multi-Density Adaptive Trajectory Clustering Algorithm for Ships Based on AIS Data

YINGJIAN ZHANG[1], XIAOYU YUAN[1], MENG LI[1], GUANG ZHAO[2], AND HONGBO WANG[1], (Member, IEEE)

[1]State Key Laboratory on Integrated Optoelectronics, College of Electronic Science and Engineering, Jilin University, Changchun 130012, China
[2]Tianjin Navigation Instrument Research Institute, Tianjin 300131, China

Corresponding author: Hongbo Wang (wang_hongbo@jlu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant u1964202.

ABSTRACT Automatic Identification System (AIS) can obtain a large amount of data on ship trajectories and movement characteristics, which provides data support for ship route planning, navigation safety, and maritime traffic control. Therefore, it is of great importance to analyze and cluster the AIS data to obtain an understanding of the movement behavior of ships. The Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm is capable of discovering arbitrarily shaped clusters and is suitable for clustering ship trajectories. However, the traditional DBSCAN algorithm suffers from the following shortcomings. First, the DBSCAN algorithm requires the input of a similarity matrix between the trajectories, and the similarity matrix obtained from the traditional Hausdorff distances does not identify the trajectory directions. Second, the DBSCAN algorithm is sensitive to the input parameters, and the clustering results obtained from different datasets with the same parameters vary widely. Finally, the DBSCAN algorithm is poor at clustering multi-density distribution datasets. To address these shortcomings, this study proposes a multi-density adaptive trajectory clustering (MDA-Traclus) algorithm, which considers the multi-density distribution trajectory dataset, adaptively determines the input parameters, and adds a trajectory direction identification mechanism to realize the clustering of trajectory clusters with different directions and different densities. Here, the classical DBSCAN algorithm, KANN-DBSCAN algorithm, and MDA-Traclus algorithm are compared and analyzed using four manually labeled trajectory datasets and a real trajectory dataset. The results show that the MDA-Traclus algorithm and KANN-DBSCAN algorithm are able to automatically determine the input parameters when clustering trajectory datasets with uniformly distributed densities and achieve the same clustering effect as the DBSCAN algorithm. When the trajectory density distribution is not uniform and the directions of the trajectory clusters are not consistent, the MDA-Traclus algorithm can effectively identify the low-density regions, differentiate the different directions of trajectories, and improve the clustering of the trajectories.

INDEX TERMS AIS data, density-based spatial clustering of applications with noise (DBSCAN) algorithm, multi-density adaptive trajectory clustering (MDA-Traclus) algorithm, trajectory clustering.

I. INTRODUCTION

International Maritime Organization (IMO) proposes the mandatory installation of Automatic Identification System (AIS) on ships [1]. AIS uses satellites and terrestrial base stations to transmit and receive vessel position, speed, heading, and other information in real time over the Very High Frequency radio frequency band. Each AIS-equipped vessel periodically broadcasts this information so that other vessels and shore-based stations in the vicinity can receive and display the data. By collecting and analyzing AIS data over a particular area of water, the navigational behavior of vessels in that water area can be studied.

With the popularization of AIS equipment, a large amount of AIS data are being collected and used for research in various areas of marine safety. For ship trajectory anomaly

The associate editor coordinating the review of this manuscript and approving it for publication was Qilian Liang.

detection research [2], [3], [4], [5], Kontopoulos et al. [6] utilize sparse historical AIS data and polynomial interpolation to extract ship routes and detect anomalies in ship trajectories. Historical ship trajectories obtained by the enhanced Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm are clustered to construct a fairway model and provide alerts for anomalous ship navigation behavior. To address the desicion-making process in the area of ship collision avoidance [7], [8], [9], Wang et al. [10] used ship size information from AIS to construct a velocity barrier cone between convex polygonal targets using the velocity barrier method. Then, they defined a new collision risk assessment model based on the relative velocities of the two targets and the geometric relationship between the velocity barrier cones. In terms of ship trajectory prediction [11], [12], [13], Suo et al. [14] extracted a series of trajectories from AIS ship data (i.e., latitude, longitude, speed, and heading), derived the main trajectories by applying the DBSCAN algorithm and finally introduced a deep learning framework and Gate Recirculation Unit model for predicting ship trajectories. In terms of ship route planning [15], [16], [17], Zhang et al. [18] proposed a shortest-path planning method based on AIS data, establishing a low-precision environment model, and determining the grid area of the shortest path through an ant colony algorithm to reduce the amount of computation and ultimately determine the optimal path under a finer environment model through the A* algorithm. Regarding traffic management [19], Xu et al. [20] proposed a framework based on AIS data analysis, including a historical traffic analysis module, K-means based attribute classification, and short-term traffic prediction module based on Back-Propagation Artificial Neural Network, to improve the efficiency of operations management in the Vessel Traffic Service. In the context of marine fisheries [21], Jiang et al. [22] proposed the use of autoencoders to automatically locate features in AIS data to detect fishing activity.

The extensive and rich data provided by AIS makes it an important data source in the field of maritime traffic and management, However, as the dataset is very large and may be affected by noise, interference, and false data, it is necessary to cluster the trajectories provided by the AIS data to identify and exclude the abnormal data, which can assist in the analysis of ship movement behavior and provide data support for ship route planning, detection of abnormal behavior at sea, and other tasks.

To perform clustering analysis of AIS data, commonly used clustering methods include the DBSCAN algorithm [23], K-means algorithm [24], Gaussian Mixture Model [25], QuickBundles [26], and Deep Embedded Clustering algorithm [27]. Among them, the DBSCAN algorithm is widely used for ship trajectory clustering because it can effectively cluster irregular datasets and is less affected by abnormal data. Li et al. [28] proposed a DBSCAN algorithm based on nearest-neighbor similarity and a fast nearest-neighbor search algorithm, which effectively improves the efficiency of DBSCAN by reducing the redundant distance calculation and incorporating a parallel distance query technique to speed up the identification of the neighbors of each point. Wang et al. [29] proposed a ship trajectory clustering algorithm based on HDBSCAN, which extends DBSCAN by transforming the DBSCAN algorithm into a hierarchical clustering algorithm that extracts a flat cluster using a stable clustering technique. Li et al. [30] used the multidimensional scaling method to compute a distance matrix between trajectories and determined the input parameters Eps and MinPts for clustering from the k-th nearest neighbor distance curve for each trajectory and the first-order derivatives of the k-th nearest neighbor distance curve. Zhang et al. [31] proposed the FA-DBSCAN algorithm using fuzzy adaptive density to cluster the turning points of the ship trajectories. The algorithm obtains the density of each sample by adaptively determining the neighborhood radius from the distance between the data samples and gradually increases the clustering center according to the density of the samples. Meanwhile, it selects the optimal number of clusters and clustering centers from using the fuzzy clustering effectiveness index. Finally, the clustering results are optimized by minimizing the clustering objective function to achieve the parameter adaptation of the DBSCAN algorithm. Han et al. [32] computed the similarity matrix between trajectories by using the Mahalanobis distance and applied a Map-Reduce method to handle problems with large data in selecting the parameters solving the problem of the DBSCAN algorithm sensitivity to the input parameters. Zhou et al. [33] proposed an adaptive DBSCAN algorithm based on the Chameleon Swarm algorithm which iteratively optimizes the clustering evaluation metrics as the optimization objective function to obtain the optimal input parameters and clustering results.

More research attention has been given to reducing the number of operations of the DBSCAN algorithm and adaptively determining the proper input parameters. However, the clustering effect of the DBSCAN algorithm is not satisfactory when faced with datasets containing multiple density distributions and different trajectory directions. Therefore, this study proposes a multi-density adaptive trajectory clustering algorithm (MDA-Traclus) based on the DBSCAN algorithm. The main contributions of the proposed algorithm are as follows. First, the algorithm adds density threshold detection based on the K-average nearest neighbors (KANN) algorithm and calculates the input parameters according to the set density threshold for the high-density region to obtain the initial clustering results. Then, secondary clustering is performed for the low-density region according to the obtained clustering results to improve the clustering accuracy of the algorithm. Finally, the Hausdorff distance calculation is improved to include the distance penalty factor, which improves the accuracy of the similarity between the trajectories effectively identifying the direction of the trajectories.

The rest of the paper is organized as follows. Section II describes the pre-processing of the AIS data. In Section III,

FIGURE 1. Trajectories retained after AIS data cleaning.

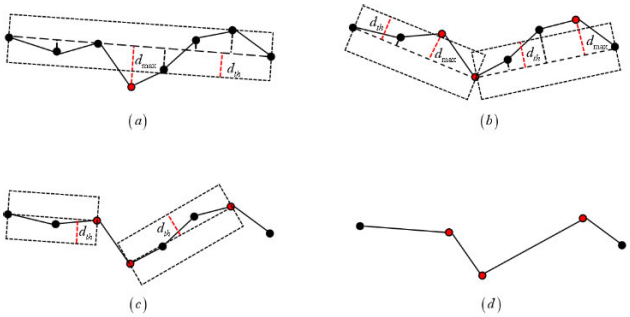


FIGURE 2. Schematic of the DP algorithm. (a)(b) Points with distances greater than the threshold are key points. (c) Points with distances less than the threshold are removed. (d) Simplified trajectory.

the principles and implementation of the classical DBSCAN algorithm and the MDA-Traclus algorithm are introduced. In Section IV, the clustering experiments using the manually labeled trajectory dataset and the real trajectory dataset are presented to compare the clustering effects of the classical DBSCAN, KANN-DBSCAN and MDA-Traclus algorithm proposed in this study. The conclusions of the study and future research directions are discussed in Section V.

II. GUIDELINES FOR MANUSCRIPT PREPARATION

AIS equipment can communicate with the surrounding ships and onshore control center and broadcast the static and dynamic AIS data from ships so as to establish the ship and shore information network. Static data include the following: name of ship, maritime mobile service identification (MMSI), ship type, ship length, and ship width. Dynamic data include the following: longitude, latitude, heading, and speed. In addition, there are also navigation data, such as cargo, draft, destination, and estimated time of arrival. In this study, we primarily utilize ship MMSI, ship length, longitude, and latitude.

A. AIS DATA CLEANING

As AIS data can be affected by factors such as noise, errors, and omissions, there are a large amount of invalid data; thus, it is necessary to process the raw data appropriately to remove the invalid data and reduce possible errors in subsequent work. Errors fall into the following categories: (1) the length of the MMSI is not 9 bits or is unreasonable; (2) the longitude is less than 0° or greater than 180°, and the latitude is less than 0° or greater than 90°; (3) the ship's heading or speed is out of range (e.g., heading is greater than 360° or speed is less than 0kn) [34]. In addition, incomplete trajectories with fewer than 20 trajectory points are also eliminated. False data not only increase the running time of the algorithm but also affect the final clustering calculation, which affects the subsequent research. After trajectory culling, 139 trajectories with a total of 42,150 trajectory points are finally retained for subsequent trajectory clustering studies. The retained trajectories are shown in Fig. 1.

B. TRAJECTORY COMPRESSION

For dynamic vessels, AIS updates its information every 2 to 12 seconds. It is evident that the update rate of vessel trajectory information is much faster than changes in actual navigation parameters such as position and heading. Consequently, AIS data contains a significant amount of redundant information, which affects the computational speed of clustering analysis.

To reduce computational costs, it is necessary to compress the raw AIS trajectory data. The Douglas-Peucker (DP) algorithm [35], [36] is a commonly used trajectory compression algorithm that reduces the number of points in a trajectory, allowing for data compression and simplification. The DP algorithm removes the redundant points on the trajectory step by step and preserves the shape of the trajectory with the following basic principles. Input a series of discrete points $O = \{p_1, p_2, \cdots, p_N\}$ of the trajectory with a set of key points K. Connect the starting point and end point of the trajectory with a straight line and calculate the distances of all points of the trajectory from the straight line. Compare the maximum distance $D_{\max}$ from the farthest point P_i with compression threshold D_{th}. If $D_{\max} < D_{th}$, remove all points between the start point and end point of the current segment and keep only the start point and end point as key points. If $D_{\max} \geq D_{th}$, add point P_i to the key points and use it as a boundary to divide the trajectory into two sub-segments, performing the same operation on each of the two sub-segments until all the sub-segments contain only the start point and end point. The set of key points obtained at that time is the compressed trajectory.

For example, Fig. 2 shows the compression process of the ship trajectories. After data cleaning, 139 ship trajectories are collected containing 42,150 trajectory points, and 2260 trajectory points are retained after compression by the DP algorithm, with the compression rate reaching 94.6%. As can be seen from Fig. 3, after compression by the DP algorithm, the shape characteristics of the ship trajectory can be well-preserved. Thus, this algorithm is chosen for ship trajectory compression.

III. MDA-TRACLUS ALGORITHM

The traditional DBSCAN algorithm often uses the Euclidean distance to compute the distance matrix for point clustering; meanwhile, the Hausdorff distance is often used to compute

the similarity matrix between trajectories in the clustering method. The traditional Hausdorff distance is easily affected by outliers and cannot distinguish the direction of trajectories when calculating the similarity between trajectories. In addition, the clustering effect is poor for datasets with uneven density distributions. Therefore, in this study, for the above problems, the traditional method is improved, and the MDA-Traclus algorithm is proposed, with the steps of the algorithm presented as follows.

A. CALCULATION OF THE TRAJECTORY SIMILARITY MATRIX

The AIS data contain a series of spatio-temporal coordinate points, and the distance between the trajectory points, which represents the spatial straight-line distance or path length, can be used to measure the spatial similarity between the points. Meanwhile, the distance between the trajectories is affected by the shape of the trajectory and spatio-temporal relationship between the trajectory points. As opposed to calculating the distance between the points, the method of calculating the distance between the trajectories usually considers the order of the trajectory points as well as the similarity in the shapes of the trajectories. The Hausdorff distance is a commonly used similarity measure between trajectories that captures the differences in the shapes between the trajectories; a smaller Hausdorff distance corresponds to higher similarity between the trajectories. The following are the steps to calculate the similarity between trajectories using the Hausdorff distance:

- Step 1: Initialize the Hausdorff distance to 0.
- Step 2: For each point a_i in the point set $traj1$:

1) The distance from a_i to all points in the point set $traj2$ is computed to obtain a set of distance values. Here, the distance function uses the Euclidean distance.
2) Add the minimum of this set of distance values to $a\_distance\,(i)$.

- Step 3: For each point b_j in point set $traj2$:

3) Calculate the distance from b_j to all points in the point set $traj1$ to obtain another set of distance values in the same way as above.
4) Add the minimum of this set of distance values to $b\_distance\,(j)$.

- Step 4: Take the maximum value of the distance list $a\_distance\,(i)$ of the point set $traj1$ as the Hausdorff distance from $traj1$ to $traj2$, denoted $h\,(traj1, traj2)$.
- Step 5: Take the maximum value of the distance list $b\_distance\,(j)$ of the point set $traj2$ as the Hausdorff distance from $traj2$ to $traj1$, denoted $h\,(traj2, traj1)$.
- Step 6: Calculate the Hausdorff distance between traj1 and traj2 using Eq. (1) as follows:

$$Hausdorff\ (traj1, traj2) = \max\left(h\,(traj1, traj2)\,, h\,(traj2, traj1)\right). \tag{1}$$

Figure 3 illustrates how the Hausdorff distance is calculated, in which the length of the red dashed line indicates

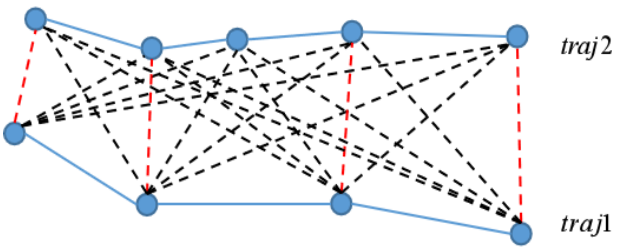


FIGURE 3. Schematic of the Hausdorff distance.

the shortest distance from each point in $traj1$ to $traj2$, and the Hausdorff distance from $traj1$ to $traj2$ is the length of the longest red dashed line.

From the above, the Hausdorff distance is evidently very sensitive to the outliers in the point set because it calculates the maximum value among the minimum values. Meanwhile, as the Hausdorff distance does not consider the order of the trajectory points when calculating the distance between the trajectories, the Hausdorff distance cannot distinguish between trajectories with opposite trajectory directions.

Because the AIS data may be affected by factors such as noise, error, and data omission, the errors or missing trajectory points may cause abrupt changes in the trajectory, causing the distance between two trajectories obtained based on the Hausdorff distance calculation to not reflect the true situation. To solve this problem, this study makes the following improvements to the Hausdorff distance.

For each path segment $Seg2$ in Trajectory 2, the distance from the path point $P1$ in Trajectory 1 is calculated to each $Seg2$. When the projection of $P1$ onto $Seg2$ falls on the trajectory segment, the distance between $P1$ and $Seg2$ equals the distance from P1 to the projection point.; if the projection point is not on the trajectory segment, the distance is the closer of the two end points of $P1$ to $Seg2$. The minimum value of the distance from $P1$ to each trajectory segment is taken as the distance from $P1$ to Trajectory 2, and the Improved-Hausdorff distance from Trajectory 1 to Trajectory 2 is calculated as the average of all the minimum values, denoted as $h\,(Traj1, Traj2)$. Similarly, the Improved-Hausdorff distance from $Traj2$ to $Traj1$ is $h\,(Traj2, Traj1)$. Finally, the Improved-Hausdorff distance is calculated as the average of $h\,(Traj1, Traj2)$ and $h\,(Traj2, Traj1)$. It is written as Eq. (2):

$$\begin{cases} h\,(Traj1, Traj2) = \dfrac{1}{N}\displaystyle\sum_{P1\in Traj1}\left(\min_{Seg2\in Traj2}\{h\,(P1, Seg2)\}\right) \\ h\,(Traj2, Traj1) = \dfrac{1}{N}\displaystyle\sum_{P2\in Traj2}\left(\min_{Seg1\in Traj1}\{h\,(P2, Seg1)\}\right) \\ H\,(Traj1, Traj2) = \dfrac{1}{2}\left(h\,(Traj1, Traj2) + h\,(Traj2, Traj1)\right), \end{cases} \tag{2}$$

where $Traj1$ and $Traj2$ denote Trajectory 1 and Trajectory 2, respectively; $P1$ and $P2$ denote the trajectory points

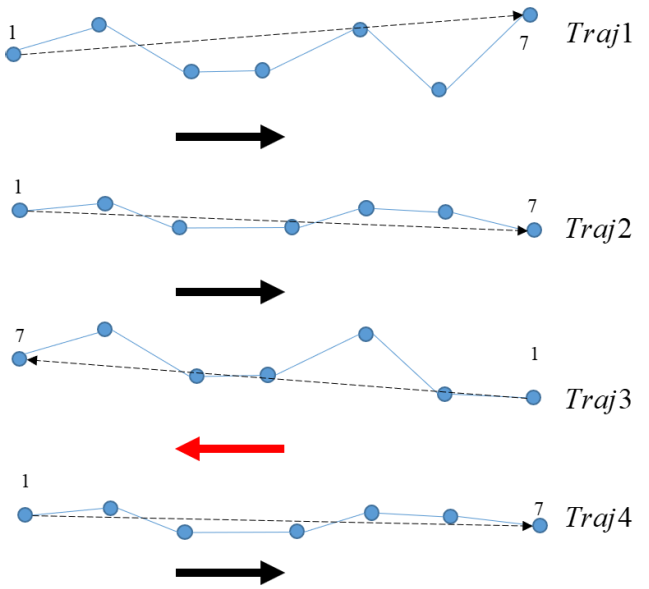


FIGURE 4. Schematic of trajectory direction.

in Trajectory 1 and Trajectory 2, respectively; *Seg*1 and *Seg*2 denote the trajectory segments in Trajectory 1 and Trajectory 2, respectively.

As the Hausdorff distance cannot identify the trajectory direction, before the inter-trajectory distance calculation, the consistency of the trajectory direction is first determined using the starting point and end point of the two trajectories to determine the starting and end points of the vector $\overrightarrow{dist_i}$. Here, the cosine value of the angle θ between the two vectors is calculated to evaluate the consistency of the direction of the two trajectories, as shown in Fig. 4.

When the cosine value of the angle between the trajectory direction vectors of *Traj*1 and *Traj*2 is greater than 0, it is regarded as the same trajectory direction, and the trajectory similarity between the two trajectories is the Improved-Hausdorff distance between the two trajectories. When the cosine value of the angle between the trajectory direction vectors of *Traj*3 and *Traj*4 is less than 0, the trajectories are considered to be in the opposite direction, and the trajectory similarity between the two trajectories is the Improved-Hausdorff distance multiplied by the distance penalty factor α to reduce the similarity between the two trajectories.

Ultimately, the Improved-Hausdorff distance is given by Eq. (3), as shown at the bottom of the next page.

B. DBSCAN ALGORITHM

DBSCAN, proposed by Ester et al. in 1996, is a density-based clustering algorithm for discovering classes or clusters in datasets with different densities [37]. Compared to traditional distance-based clustering algorithms (e.g., K-means), DBSCAN can efficiently handle datasets with irregular shapes and noise. The DBSCAN algorithm for trajectory clustering has the following definitions:

Algorithm 1 DBSCAN

Input: Similarity matrix between trajectories D, Neighborhood radius *Eps*, Minimum number of trajectories to form a dense region *Minlines*
Output: Mapping of trajectories to their respective clusters *Cluster*, Set of noise trajectories *Noise*

```
function DBSCAN(D, Eps, Minlines):
C = 0 //Cluster counter
Initialize Clusters as an empty mapping from trajectories to clusters
Initialize Noise as an empty set
for each unvisited trajectory T in D do
    If T is not visited then
        mark T as visited
        Neighbors ← getNeighbors(T, D, Eps)
        if size(Neighbors) < Minlines then
            add T to Noise
        else
            C = C + 1
            Cluster ← expandCluster(T, Neighbors, C, D, Eps, Minlines)
            Clusters[T] ← Cluster
        end if
    end if
end for
Return Cluster, Noise
```

Core Trajectory: a trajectory is considered to be a core trajectory if it has at least the minimum number *Minlines* of trajectories within a specified radius *Eps*.

Directly Density-Reachable: For two trajectories A and B, if trajectory B is in the *Eps* neighborhood of trajectory A and trajectory A is a core trajectory, then trajectory B is considered to be directly density-reachable from trajectory A.

Density-Reachable: For two trajectories A and B, B is density-reachable from A if there exists a series of core trajectories $L_1, L_2, \cdots, L_n$, where $L_1 = A$, $L_n = B$, and each L_{i+1} is directly density-reachable from L_i. That is, trajectory B can be connected to trajectory A by a series of core trajectories.

Density-Connected: Trajectories A and B are considered to be density-connected if there exists a core trajectory C such that both A and B are density-accessible from C.

Based on the above core concepts, the DBSCAN algorithm calculates the number of trajectories in its neighborhood by traversing each trajectory in the trajectory dataset, and if the trajectory is greater than or equal to *Minlines*, it is marked as a core trajectory. A cluster of trajectories is formed by gradually expanding from the unvisited core trajectories through the directly density-reachable relation and the density-reachable relation. Until all trajectories are visited, non-core trajectories that do not belong to any cluster are marked as noise trajectories.

Algorithm 2 expandCluster

Input: T, *Neighbors*, C, D, *Eps*, *Minlines*
Output: *Neighbors*
functionexpandCluster(T, *Neighbors*, C, D, *Eps*, *Minlines*):
Cluster ← {T}
for each trajectory U in *Neighbors* do
  if U is not visited then
    mark U as visited
      Neighbors_U ← getNeighbors(U, D, *Eps*)
    if *size* (*Neighbors_U*) ≥ *Minlines* then
      Neighbors ← Neighbors union *Neighbors_U*
    end if
  end if
  if U does not belong to any cluster then
    add U to cluster
  end if
end for
Return *Neighbors*

Algorithm 3 getNeighbors

Input: T, D, *Eps*
Output: *Neighbors*
function getNeighbors(T, D, *Eps*):
Neighbors ← an empty set
for each trajectory U in D do
  if $D[T][U] \geq Eps$ then
    add U to *Neighbors*
  end if
end for
Return *Neighbors*

C. DETERMINING THE PARAMETERS OF THE DBSCAN ALGORITHM

From the above, the DBSCAN algorithm has two manually entered parameters: neighborhood radius *Eps* and minimum density *MinPts*. As the DBSCAN algorithm is very sensitive to the parameters, the same input parameters yield different clustering results when encountering different datasets; therefore, how to adaptively determine clustering parameters is of great importance. To adaptively find the optimal parameter pair (*Eps*, *MinPts*) that can be set for improved clustering, Wenjie et al. [38] used the KANN algorithm to generate *Eps* candidate lists, which are used to calculates the corresponding mathematical expectations to obtain *MinPts* candidate lists, considering that each *Eps* corresponds to many *MinPts*. The *Eps* candidate list and *MinPts* candidate list are then combined into a list of parameter pairs, which are set sequentially in pairs as parameters of DBSCAN, and the number of clusters is observed. When the number of trajectory clusters remains constant after setting three consecutive parameter pairs, the algorithm sets the first set of parameters as the final clustering input parameters.

By determining the parameters through the KANN algorithm, the density concentration region can be effectively identified and divided into clusters. However, the clustering performance is poor in the face of dataset with uneven density distribution, and regions with smaller density cannot be effectively identified. Thus, this study proposes the MDA-Traclus algorithm to address this drawback, and the main steps are as follows.

1) SETTING THE DENSITY THRESHOLD

The density threshold *Density_th* is set according to the trajectory density distribution, *Density_th* = 0.3 in this study. More trajectories with high-density distribution in the dataset correspond to smaller *Density_th*, where the range of *Density_th* is [0 , 1).

2) GENERATE EPS CANDIDATE LIST

- Calculate the inter-trajectory similarity matrix, as shown in Eq. (4):

$$D_{n\times n} = \{D\_Hausdorff\ (i,j)\,|1 \leq i,j \leq n\}, \quad (4)$$

  where $D_{n\times n}$ is a real $n \times n$ symmetric matrix, n is the number of trajectories in the dataset, and *D_Hausdorff* (i,j) is the Hausdorff distance between the i-th and j-th trajectory.
- The elements of each row of the distance matrix $D_{n\times n}$ are sorted in ascending order to obtain the matrix *D_K*, and the distance vector D_0 consisting of the elements of the first column of *D_K* represents the distance of each trajectory to itself, which is 0. The elements of the K-th column are the K nearest-neighbor distance vectors $\overrightarrow{D_K}$ of all trajectories.

$$\begin{cases} h(Traj1, Traj2) = \frac{1}{N}\sum\limits_{P1\in Traj1}\left(\min\limits_{Seg2\in Traj2}\{h(P1, Seg2)\}\right) \\ h(Traj2, Traj1) = \frac{1}{N}\sum\limits_{P2\in Traj2}\left(\min\limits_{Seg1\in Traj1}\{h(P2, Seg1)\}\right) \end{cases} \cos\theta > 0$$

$$\begin{cases} h(Traj1, Traj2) = \frac{\alpha}{N}\sum\limits_{P1\in Traj1}\left(\min\limits_{Seg2\in Traj2}\{h(P1, Seg2)\}\right) \\ h(Traj1, Traj1) = \frac{\alpha}{N}\sum\limits_{P2\in Traj2}\left(\min\limits_{Seg1\in Traj1}\{h(P2, Seg1)\}\right) \end{cases} \cos\theta \leq 0. \quad (3)$$

- The K-average nearest-neighbor distance D_K of a trajectory is obtained by averaging the elements in vector $\overrightarrow{D_K}$. In the case where the trajectory density is not uniformly distributed, only the first N rows are calculated when averaging $\overrightarrow{D_K}$, as shown in Eq. (5):

$$N = \lfloor n \times (1 - Density) \rfloor, \tag{5}$$

where N is the result after rounding. As N as the number of rows must be an integer, the result of the calculation is rounded down.

This is conducted such that the parameters chosen for the first clustering are only for regions with high-density distribution. For each value of K, the candidate list is $Eps\_K\ (K)$, denoted as Eq. (6):

$$Eps\_K(K) = \frac{1}{N} \sum_{i=1}^{N} \overrightarrow{D_{Ki}}, \tag{6}$$

where $\overrightarrow{D_{Ki}}$ is the i-th element of the vector $\overrightarrow{D_K}$.

3) GENERATE MINLINES CANDIDATE LIST

For the *Eps* parameter candidate lists Eps_K of size $1 \times n$, the number of trajectories are computed corresponding to each trajectory within each *Eps* neighborhood to obtain the matrix $Minlines\_K$ of size $n \times n$. For K rows of $Minlines\_K$, the mathematical expectation is determined to obtain the candidate list $Minlines\_KK\ (K)$, written as Eq. (7):

$$Minlines\_KK\ (K) = \frac{1}{n} \sum_{i=1}^{n} Minlines\_K\ (i, K). \tag{7}$$

4) DETERMINATION OF OPTIMAL PARAMETERS

From step 2) and step 3), the *Eps* and *Minlines* candidate matrix of size $1 \times n$ can be obtained, and the candidate values are sequentially selected as input parameters of the DBSCAN algorithm for clustering, yielding the clustering results corresponding to different K values. When the number of clusters is the same for three or more consecutive runs, the number of clusters N is considered to be stable, and clustering is continued; when the number of clusters N of $K+1$ changes, the *Eps* and *Minlines* corresponding to K are considered to be the optimal values.

5) MULTI-DENSITY ADAPTIVE PARAMETERS

After performing clustering with the parameters determined in step 4), the noise trajectories in the clustering results are extracted for density detection, as shown in Eq. (8):

$$\begin{cases} \text{Secondary clustering} & noise\_traj \geq n \times Density\_th \\ \text{Output clustering result} & noise\_traj < n \times Density\_th, \end{cases} \tag{8}$$

where $noise\_traj$ is the number of noise trajectories in the first clustering result, n is the total number of trajectories in the dataset, and $Density\_th$ is the density threshold.

If secondary clustering is required, the similarity matrix D_noise of the noise trajectories is extracted, steps 2, 3 and 4 are repeated, and the results obtained from the two clusters are combined as the final clustering result.

D. INDICATORS TO EVALUATE CLUSTERING RESULTS

When performing cluster analysis, evaluation metrics can help evaluate the quality and effectiveness of the different clustering results. Clustering metrics are usually divided into external and internal evaluation methods, where external evaluation methods refer to the evaluation of the goodness of clustering results, such as the Purity, Randall's coefficient, and F-value, in which the real labels in the dataset are known. Meanwhile, internal evaluation methods refer to the evaluation of the results of clustering results based only on the clustering results, such as the contour coefficient, without the help of external information. In this study, the clustering analysis of the AIS data of the Yangtze River waters cannot mark the real labels of the trajectories; thus, the internal contour coefficient metric is adopted to evaluate the clustering performance.

The contour coefficient is a measure of the similarity of a sample to other samples in the cluster to which it belongs versus the dissimilarity with samples from the nearest clusters. It is given in Eq. (9):

$$silhouette\_score = \frac{1}{n} \sum_{i=1}^{n} \frac{\sigma\ (i) - \mu\ (i)}{\max\ \{\mu\ (i)\ , \sigma\ (i)\}}, \tag{9}$$

where $\mu\ (i)$ is the average distance of sample i from other samples in the same cluster, indicating the density within the cluster to which it belongs; $\sigma\ (i)$ is the average distance of sample i from the nearest other clusters, indicating the degree of separation of sample i from the other clusters; and $silhouette\_score$ is the average of the profile coefficients of all samples, which is used as a measure of the quality of the entire cluster.

The value of $silhouette\_score$ ranges from $[-1, 1]$, in which values closer to 1, correspond to more reasonable clustering results and better separation between the clusters. In general, a better clustering algorithm can produce results with an average profile coefficient near 0.5 or higher.

IV. RESULTS

To verify the effectiveness of the algorithm, we compared the MDA-Traclus algorithm with the traditional DBSCAN algorithm and KANN-DBSCAN algorithm to verify the clustering effect of the three algorithms when encountering different types of trajectory datasets. The algorithms were run on Windows 64-bit with eight cores (Intel(R) Core (TM) i7- 11700 CPU @ 2.5GHz to 4.9GHz) and 16GB of RAM using the MATLAB R2022a implementation.

A. DBSCAN ALGORITHM

As there is no dataset specifically used to test the effect of trajectory clustering, four trajectory datasets are manually

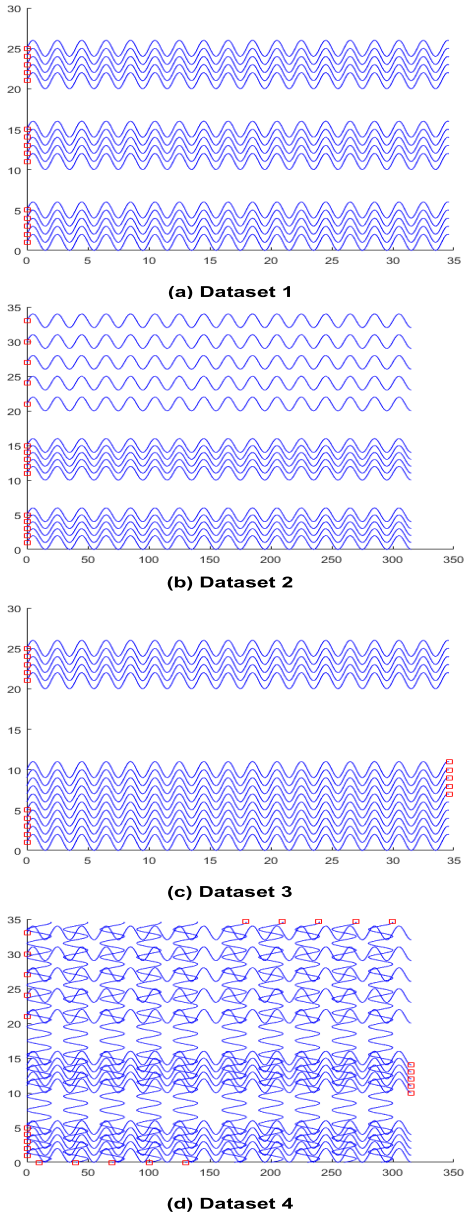


FIGURE 5. Manually labeled trajectory dataset.

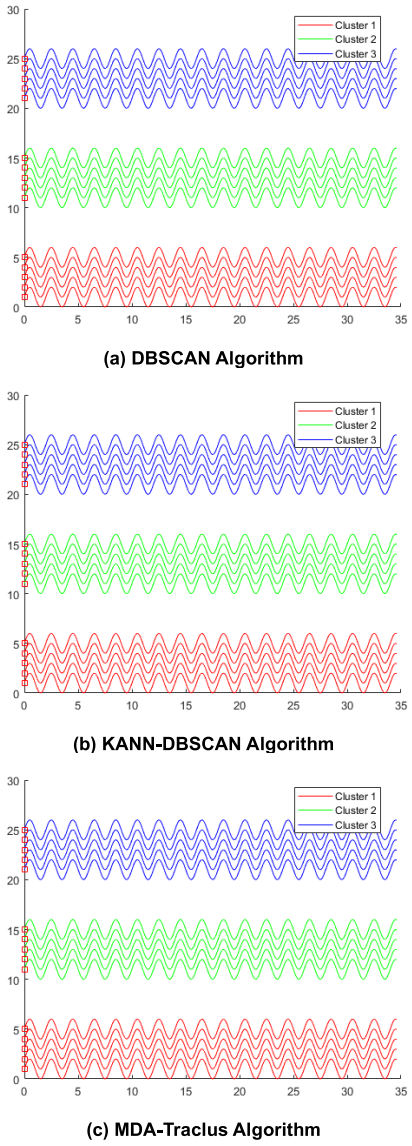


FIGURE 6. Dataset 1 clustering results.

TABLE 1. Input parameters of the three algorithms for Dataset 1.

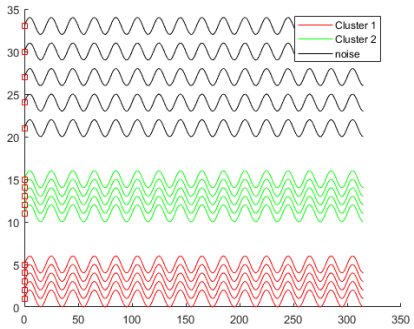
Algorithm	DBSCAN	KANN-DBSCAN	MDA-Traclus
Eps	4.0	3.2	2.4595
Minlines	5.0	4.6	4.6

assembled to test the performance of the trajectory clustering algorithm. The test trajectory datasets used in this study are shown in Fig. 5.

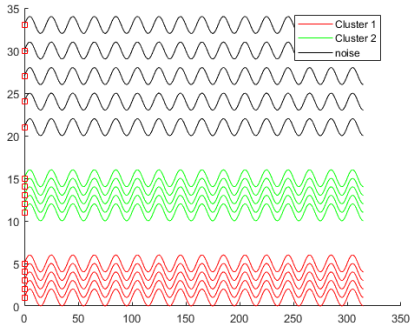
Fig. 5 shows four sets of trajectory data with different distributions, where the red squares indicate the trajectory starting points. Fig. 5 (a) shows three sets of trajectory datasets with the same density distribution; Fig. 5 (b) illustrates three sets of trajectory datasets with different density distributions; Fig. 5 (c) shows three sets of trajectory datasets with the same density distribution, where one set of trajectories has a trajectory direction opposite to those of the other trajectories; and Fig. 5 (d) illustrates five sets of trajectory datasets with different density distributions and different trajectory directions, where two sets of trajectories intersect the other three sets of trajectories. These four types of trajectories are used to verify the effectiveness of the MDA-Traclus algorithm by comparing the traditional DBSCAN algorithm and KANN-DBSCAN algorithm.

For Dataset 1, the input parameters for the three algorithms are shown in Table 1.

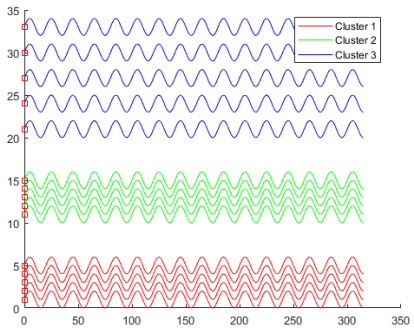
Fig. 6 shows the clustering results of the three algorithms for Dataset 1. For Dataset 1 containing trajectories of the



(a) DBSCAN Algorithm



(b) KANN-DBSCAN Algorithm



(c) MDA-Traclus Algorithm

FIGURE 7. Dataset 2 clustering results.

same density distribution, all three algorithms accurately cluster the data into three trajectory clusters.

Table 1 shows the input parameters of the three algorithms, where the input parameters of the DBSCAN algorithm are determined empirically and by multiple experimental trials, and the input parameters of the KANN-DBSCAN and MDA-Traclus algorithms are determined adaptively by the algorithm.

For Dataset 2, the input parameters for the three algorithms are shown in Table 2:

Fig. 7 shows the clustering results of the three algorithms for Dataset 2, containing trajectory datasets with different density distributions. Here, the DBSCAN and KANN-DBSCAN algorithms cannot effectively identify the trajectory clusters from the low-density region, which are processed as noise trajectories. The first clustering of the

TABLE 2. Input parameters of the three algorithms for Dataset 2.

Algorithm	DBSCAN	KANN-DBSCAN	MDA-Traclus-i	MDA-Traclus-ii
Eps	5.0	5.0	4.885	9.3798
Minlines	5.0	4.2	4.2	4.6

TABLE 3. Input parameters of the three algorithms for Dataset 3.

Algorithm	DBSCAN	KANN-DBSCAN	MDA-Traclus
Eps	10.0	2.8	2.4601
Minlines	5.0	4.2	4.6

TABLE 4. Input parameters of the three algorithms for Dataset 4.

Algorithm	DBSCAN	KANN-DBSCAN	MDA-Traclus-i	MDA-Traclus-ii
Eps	5.0	35.8	4.6924	24.2978
Minlines	3.0	10.04	2.92	2.6

MDA-Traclus algorithm to determine the input parameters for the high-density region accurately identify Clusters 1 and 2. After the density detection, a second clustering is determined to be necessary, identifying the low-density region Clusters 3. Finally, the results of the two clusters are merged, and Dataset 2 was clustered into three trajectory clusters.

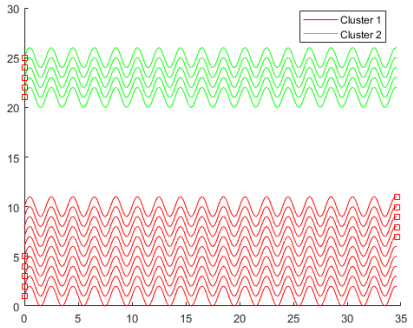
Table 2 shows the input parameters of the three algorithms for Dataset 2. The input parameters of the DBSCAN algorithm are determined empirically and through multiple experimental trials, while the input parameters of the KANN-DBSCAN and MDA-Traclus algorithms are determined adaptively by the algorithm. MDA-Traclus-i is the set of input parameters determined by the first clustering, and MDA-Traclus-ii is the set of input parameters determined for the second clustering.

Table 3 shows the input parameters of the three algorithms for Dataset 3, determined as above.

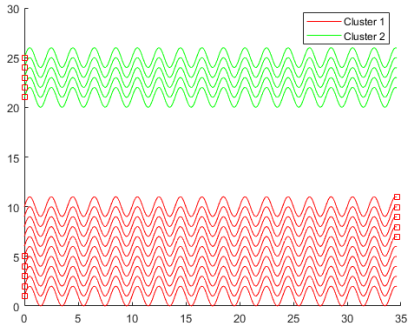
Fig. 8 shows the clustering results of the three algorithms for Dataset 3. For the dataset of trajectories distributed in different directions, the DBSCAN and KANN-DBSCAN algorithms are unable to detect the trajectory direction because they use the traditional Hausdorff distance and, therefore, incorrectly classify trajectories that are adjacent to each other but have opposite trajectory directions into one cluster. The MDA-Traclus algorithm uses the Impoved-Hausdorff distance, which can effectively detect the trajectory direction and accurately cluster Dataset 3 into three trajectory clusters.

Table 4 shows the input parameters of the three algorithms for Dataset 4, determined as above.

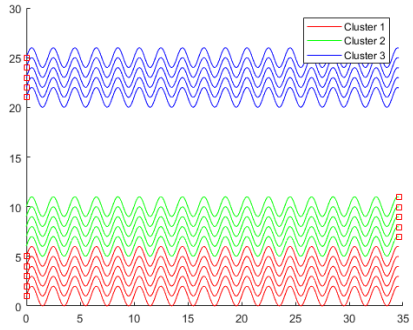
Fig. 9 shows the clustering results of the three algorithms for Dataset 4, containing trajectories with different densities and different directional distributions. As can be seen from the figure, owing to the further widening of the density gap



(a) DBSCAN Algorithm

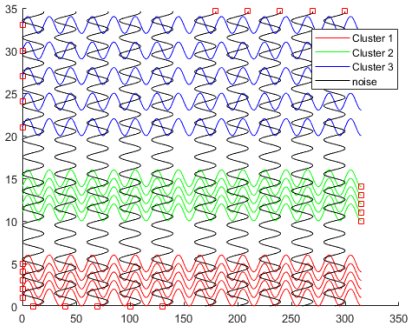


(b) KANN-DBSCAN Algorithm

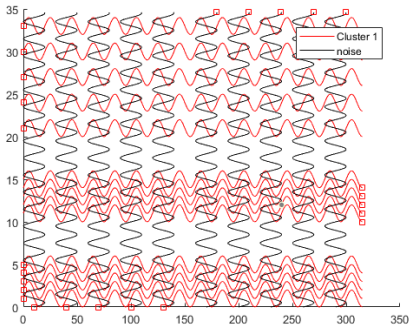


(c) MDA-Traclus Algorithm

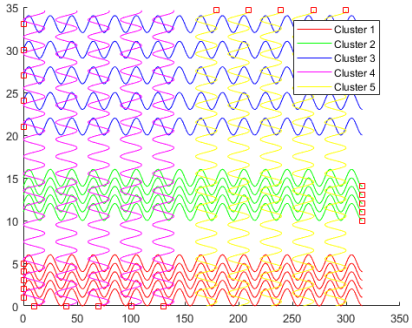
FIGURE 8. Dataset 3 clustering results.



(a) DBSCAN Algorithm



(b) KANN-DBSCAN Algorithm



(c) MDA-Traclus Algorithm

FIGURE 9. Dataset 4 clustering results.

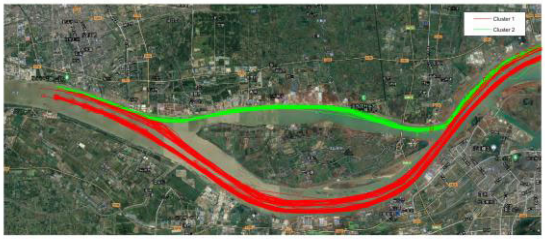
in the trajectory dataset, the DBSCAN algorithm is only able to identify three trajectory clusters in the high-density region, and the two trajectory clusters in the low-density region are treated as noise trajectories. The KANN-DBSCAN algorithm, owing to the large gap between the density distributions of the dataset, is unable to correctly identify trajectory clusters at all, clustering the trajectories of the high-density region as a single cluster, and the trajectories of the low-density region are considered as noise trajectories. The MDA-Traclus algorithm first clusters the high-density regions to determine the input parameters, accurately identifying Cluster 1, 2, and 3. After the density detection, the second clustering is performed to identify low-density regions Cluster 4, and 5 and can correctly distinguish between different trajectory directions. Finally, the two clustering results are merged to cluster Dataset 4 into five trajectory clusters.

FIGURE 10. *U* pstream trajectory of the Shiyezhou section.

B. REAL SCENARIO VALIDATION

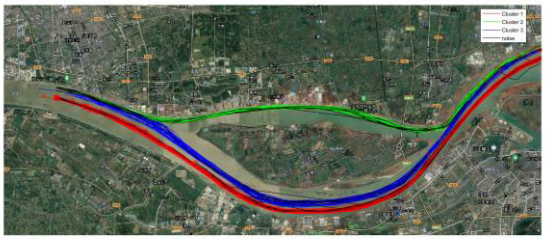
The evaluation of real trajectories is performed by using the AIS trajectory data of the Shiyezhou section of the Yangtze River waters after the processing described in Section II. There are three channels in this section, including two

FIGURE 11. *D* ownstream trajectory of the Shiyezhou section.



(a) DBSCAN Algorithm

(b) KANN-DBSCAN Algorithm



(c) MDA-Traclus Algorithm

FIGURE 12. Real trajectory clustering results.

upstream channels, as shown in Fig. 10 and one downstream channel shown in Fig. 11.

Table 5 shows the input parameters of the three algorithms for the real trajectory. The input parameters of the DBSCAN algorithm are determined empirically and through multiple experimental trials, while the input parameters of the KANN-DBSCAN and MDA-Traclus algorithms are determined adaptively by the algorithm. MDA-Traclus-i and MDA-Traclus-ii respectively represent the input parameters determined for the first and second clustering.

Fig.12 (a) shows the clustering results of the DBSCAN algorithm on real trajectories. As it uses the traditional Hausdorff distance to calculate the similarity between trajectories, it cannot differentiate between the upstream and downstream trajectories, and there are no indicated noise trajectories.

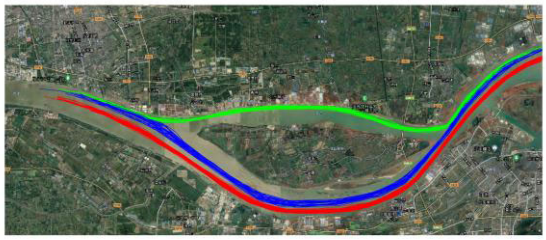


FIGURE 13. Trajectory clusters obtained by the MDA-Traclus algorithm.

TABLE 5. Real trajectory input parameters of the three algorithms.

Algorithm	DBSCAN	KANN-DBSCAN	MDA-Traclus-i	MDA-Traclus-ii
Eps	150.0	9.7359	7.1199	15.1995
Minlines	5.0	9.4029	5.4964	8.9434

TABLE 6. Clustering evaluation metrics for three algorithms.

Algorithm	DBSCAN	KANN-DBSCAN	MDA-Traclus
silhouette_score	0.3852	0.3925	0.5879

From Fig. 12 (b), we can see that the KANN-DBSCAN algorithm also cannot discriminate between the upstream and downstream trajectories, and there are 29 noise trajectories indicated. As shown in Fig. 12 (c), the MDA-Traclus algorithm can effectively discriminate between the upstream and downstream trajectories and automatically identify high-density distribution areas. Here, 139 trajectories are classified into three trajectory clusters by twice clustering, in which Cluster 1 are the downstream trajectories, Cluster 2 and Cluster 3 are the upstream trajectories, and there are 25 noise trajectories, trajectories that clearly deviate from the channels. The final clusters of trajectories are extracted as shown in Fig. 13.

The performance of three clustering algorithms on real trajectories is shown in Table 6, evaluated by metrics computed according to Eq. (9). The contour coefficients of the DBSCAN algorithm and KANN-DBSCAN algorithm are similar; however, both are less than 0.5, and the clustering is not sufficient. The clustering results obtained by the MDA-Traclus algorithm have contour coefficients that are clearly higher than 0.5, demonstrating that the algorithm proposed in this study exhibits better clustering performance.

From the above experimental results, it can be seen that the MDA-Traclus algorithm can adaptively determine the input parameters and effectively detect the noise trajectories when faced with a trajectory dataset with multiple densities and different direction distributions. Using real trajectories show clustering performance that is significantly better than that of the traditional DBSCAN algorithm and KANN-DBSCAN algorithm.

V. CONCLUSION

AIS provides huge amounts of data for purposes such as ship route planning and navigation safety. Clustering analysis of massive amounts of trajectory data is of great significance for obtaining ship behavior information, identifying collision risk, and predicting future ship demand. Therefore, this study conducts trajectory clustering analysis based on AIS data and proposes an improved algorithm referred to as MDA-Traclus to overcome the shortcomings of the DBSCAN algorithm, which is manifested in the difficulty in determining input parameters, poor effect of clustering on trajectories with multi-density distributions, and difficulty in recognizing the direction of the trajectory in traditional similarity computation methods. The improvements include the following:

- The traditional Hausdorff distance calculation method is improved and trajectory direction detection is added, which can effectively detect upstream and downstream trajectories.
- The calculation method of the input parameter *Eps* of the KANN-DBSCAN algorithm is improved for the effective identification of trajectories in high-density distribution regions.
- The density detection mechanism is added to perform secondary clustering for low-density distribution trajectories, which effectively solves the problem that a large number of low-density distribution trajectories are falsely detected as noise trajectories when the trajectory distribution is not uniform.

As shown by the experimental results of the manually labeled trajectory dataset and the real trajectory dataset, the MDA-Traclus algorithm can adaptively determine the input parameters in the face of different trajectory datasets, significantly reduce the time needed to select the input parameters, effectively identify noise trajectories, and cluster upstream and downstream trajectories in inland waters with good results. Therefore, in marine transportation research and related marine fields, the MDA-Traclus algorithm can be used for trajectory clustering with better clustering results. In our future research work, based on the clustering results of the MDA-Traclus algorithm on AIS data for inland waterways, we will extract the usual inland waterways, develop a route-planning algorithm that meets the constraints of the inland waterways, and provide more support for vessel traffic monitoring and other aspects. Meanwhile, owing to the calculation method of input parameters and similarity between trajectories, the computational efficiency will be reduced when the amount of trajectory data is too large, and we will conduct further research on the computational speed of the DBSCAN algorithm to reduce the computation time and improve the computational efficiency in the clustering and practical application of AIS data.

REFERENCES

[1] *International Convention for the Safety of Life at Sea (SOLAS)*, IMO, London, U.K., 1974.

[2] B. H. Soleimani, E. N. De Souza, C. Hilliard, and S. Matwin, "Anomaly detection in maritime data based on geometrical analysis of trajectories," in *Proc. 18th Int. Conf. Inf. Fusion*, Washington, DC, USA, Jul. 2015, pp. 1100–1105.

[3] A. McAbee, J. Scrofani, M. Tummala, D. Garren, and J. McEachen, "Traffic pattern detection using the Hough transformation for anomaly detection to improve maritime domain awareness," in *Proc. 17th Int. Conf. Inf. Fusion (FUSION)*, Jul. 2014, pp. 1–6.

[4] P.-R. Lei, "A framework for anomaly detection in maritime trajectory behavior," *Knowl. Inf. Syst.*, vol. 47, no. 1, pp. 189–214, Apr. 2016.

[5] D. Nguyen, R. Vadaine, G. Hajduch, R. Garello, and R. Fablet, "GeoTrackNet—A maritime anomaly detector using probabilistic neural network representation of AIS tracks and a contrario detection," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 6, pp. 5655–5667, Jun. 2022.

[6] I. Kontopoulos, I. Varlamis, and K. Tserpes, "A distributed framework for extracting maritime traffic patterns," *Int. J. Geographical Inf. Sci.*, vol. 35, no. 4, pp. 767–792, Apr. 2021.

[7] J.-H. Shi and Z.-J. Liu, "Deep learning in unmanned surface vehicles collision-avoidance pattern based on AIS big data with double GRU-RNN," *J. Mar. Sci. Eng.*, vol. 8, no. 9, p. 682, Sep. 2020.

[8] R. Zheng, Q. Zhou, and C. Wang, "Inland river ship auxiliary collision avoidance system," in *Proc. 18th Int. Symp. Distrib. Comput. Appl. Bus. Eng. Sci. (DCABES)*, Nov. 2019, pp. 56–59.

[9] M. Abebe, Y. Noh, Y. J. Kang, C. Seo, D. Kim, and J. Seo, "Ship trajectory planning for collision avoidance using hybrid ARIMA-LSTM models," *Ocean Eng.*, vol. 256, Jul. 2022, Art. no. 111527.

[10] Y. Wang, Y. Zhang, H. Zhao, and H. Wang, "Assessment method based on AIS data combining the velocity obstacle method and Pareto selection for the collision risk of inland ships," *J. Mar. Sci. Eng.*, vol. 10, no. 11, p. 1723, Nov. 2022.

[11] Y. Sun, X. Chen, L. Jun, J. Zhao, Q. Hu, X. Fang, and Y. Yan, "Ship trajectory cleansing and prediction with historical AIS data using an ensemble ANN framework," *Int. J. Innov. Comput. Inf. Control*, vol. 17, no. 2, pp. 443–459, Apr. 2021.

[12] S. Guo, H. Zhang, and Y. Guo, "Toward multimodal vessel trajectory prediction by modeling the distribution of modes," *Ocean Eng.*, vol. 282, Aug. 2023, Art. no. 115020.

[13] J. Chen, J. Zhang, H. Chen, Y. Zhao, and H. Wang, "A TDV attention-based BiGRU network for AIS-based vessel trajectory prediction," *iScience*, vol. 26, no. 4, Apr. 2023, Art. no. 106383.

[14] Y. Suo, W. Chen, C. Claramunt, and S. Yang, "A ship trajectory prediction framework based on a recurrent neural network," *Sensors*, vol. 20, no. 18, p. 5133, Sep. 2020.

[15] Y. K. He, D. Zhang, J. Zhang, M. Y. Zhang, and T. W. Li, "Ship route planning using historical trajectories derived from AIS data," *TransNav, Int. J. Mar. Navigat. Saf. Sea Transp.*, vol. 13, no. 1, pp. 69–76, 2019.

[16] S. O. Onyango, S. A. Owiredu, K.-I. Kim, and S.-L. Yoo, "A quasi-intelligent maritime route extraction from AIS data," *Sensors*, vol. 22, no. 22, p. 8639, Nov. 2022.

[17] P. Han and X. Yang, "Big data-driven automatic generation of ship route planning in complex maritime environments," *Acta Oceanologica Sinica*, vol. 39, no. 8, pp. 113–120, Aug. 2020.

[18] Y. Zhang, Y. Wen, and H. Tu, "A method for ship route planning fusing the ant colony algorithm and the A* search algorithm," *IEEE Access*, vol. 11, pp. 15109–15118, 2023.

[19] S. Ichimura and Q. Zhao, "Route-based ship classification," in *Proc. IEEE 10th Int. Conf. Awareness Sci. Technol. (iCAST)*, Oct. 2019, pp. 1–6.

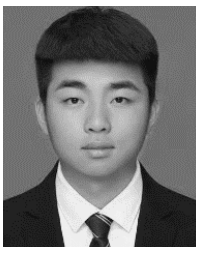
[20] G. Xu, F. Li, and C.-H. Chen, "Local AIS data analytics for efficient operation management in vessel traffic service," in *Proc. 13th IEEE Conf. Autom. Sci. Eng. (CASE)*, Aug. 2017, pp. 1668–1672.

[21] Y. Wang, S. Xu, and Z.-Z. Wang, "Design and application of AIS technology in marine fishing management," in *Proc. 2nd Int. Conf. Teaching Comput. Sci.*, 2014, pp. 136–138.

[22] X. Jiang, D. L. Silver, B. F. Hu, E. N. de Souza, and S. Matwin, "Fishing activity detection from AIS data using autoencoders," in *Proc. 29th Canadian Conf. Artif. Intell. (AI)* (Lecture Notes in Artificial Intelligence), vol. 9673, 2016, pp. 33–39.

[23] J. Chen, H. Chen, Q. Chen, X. Song, and H. Wang, "Vessel sailing route extraction and analysis from satellite-based AIS data using density clustering and probability algorithms," *Ocean Eng.*, vol. 280, Jul. 2023, Art. no. 114627.

[24] M. Mieczynska and I. Czarnowski, "Impact of distance measures on the performance of AIS data clustering," *Comput. Syst. Sci. Eng.*, vol. 36, no. 1, pp. 69–82, Jan. 2021.
[25] B. Hu, R. W. Liu, K. Wang, Y. Li, M. Liang, H. Li, and J. Liu, "Statistical analysis of massive AIS trajectories using Gaussian mixture models," in *Proc. 2nd Int. Conf. Multimedia Image Process. (ICMIP)*, Mar. 2017, pp. 113–117.
[26] T. Guo and L. Xie, "Research on ship trajectory classification based on a deep convolutional neural network," *J. Mar. Sci. Eng.*, vol. 10, no. 5, p. 568, Apr. 2022.
[27] L. Xiong, X. Xiong, F. Zhang, and H. Chen, "Unsupervised deep embedding clustering for AIS trajectory," in *Proc. IEEE Int. Geosci. Remote Sens. Symp.*, Jul. 2022, pp. 2283–2286.
[28] S.-S. Li, "An improved DBSCAN algorithm based on the neighbor similarity and fast nearest neighbor query," *IEEE Access*, vol. 8, pp. 47468–47476, 2020.
[29] L. Wang, P. Chen, L. Chen, and J. Mou, "Ship AIS trajectory clustering: An HDBSCAN-based approach," *J. Mar. Sci. Eng.*, vol. 9, no. 6, p. 566, May 2021.
[30] H. Li, J. Liu, K. Wu, Z. Yang, R. W. Liu, and N. Xiong, "Spatio-temporal vessel trajectory clustering based on data mapping and density," *IEEE Access*, vol. 6, pp. 58939–58954, 2018.
[31] D. Zhang, Y. Zhang, and C. Zhang, "Data mining approach for automatic ship-route design for coastal seas using AIS trajectory clustering analysis," *Ocean Eng.*, vol. 236, Sep. 2021, Art. no. 109535.
[32] X. Han, C. Armenakis, and M. Jadidi, "Modeling vessel behaviours by clustering AIS data using optimized DBSCAN," *Sustainability*, vol. 13, no. 15, p. 8162, Jul. 2021.
[33] W. Zhou, L. Wang, X. Han, Y. Wang, Y. Zhang, and Z. Jia, "Adaptive density spatial clustering method fusing chameleon swarm algorithm," *Entropy*, vol. 25, no. 5, p. 782, May 2023.
[34] J. Yang, Y. Liu, L. Ma, and C. Ji, "Maritime traffic flow clustering analysis by density based trajectory clustering with noise," *Ocean Eng.*, vol. 249, Apr. 2022, Art. no. 111001.
[35] D. H. Douglas and T. K. Peucker, "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature," *Cartographica, Int. J. Geographic Inf. Geovisualization*, vol. 10, no. 2, pp. 112–122, Dec. 1973.
[36] Z. Zhou, Y. Zhang, X. Yuan, and H. Wang, "Compressing AIS trajectory data based on the multi-objective peak Douglas–Peucker algorithm," *IEEE Access*, vol. 11, pp. 6802–6821, 2023.
[37] M. Ester, H. P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. KDD*, 1996, pp. 226–231.
[38] L. I. Wenjie, Y. Shiqiang, J. Ying, Z. Songzhi, and W. Chengliang, "Research on method of self-adaptive determination of DBSCAN algorithm parameters," *Comput. Eng. Appl.*, vol. 55, no. 5, pp. 1–7, 2019.



YINGJIAN ZHANG was born in Chifeng, China, in 1998. He received the bachelor's degree in electronic information science and technology from Jilin University, Jilin, in 2020, where he is currently pursuing the M.D. degree in circuits and systems. His research interest includes ship route planning.

XIAOYU YUAN was born in Taiyuan, China, in 2000. He received the bachelor's degree in electronic information engineering from Qinghai Normal University, Qinghai, in 2022. He is currently pursuing the M.D. degree in circuits and systems with Jilin University. His research interest includes ship route planning.

MENG LI was born in Inner Mongolia, China, in 2000. He received the B.Sc. degree in electronic information science and technology from Jilin University, in 2023, where he is currently pursuing the M.E. degree in integrated circuit engineering. His research interests include ship motion control and SLAM perception technology.

GUANG ZHAO was born in Shanxi, China, in 1982. He received the master's degree from the China Ship Research and Development Academy, China. Currently, he is with the Tianjin Navigation Instrument Research Institute. His research interests include ship motion control, weather routing, and ship collision avoidance.

HONGBO WANG (Member, IEEE) was born in Changchun, China, in 1969. She received the Ph.D. degree from Saint Petersburg State University, Russia. She is currently with the College of Electronic Science and Engineering, Jilin University. Her research interests include optimal control, weather routing, collision avoidance, and environmental perception.