*The Journal of Navigation* (2024), 1–22
doi:10.1017/S0373463324000171

**RESEARCH ARTICLE**

# An online method for ship trajectory compression using AIS data

Zhao Liu,[1,2] Wensen Yuan,[1] Maohan Liang,[1] Mingyang Zhang,[3]* Cong Liu,[3] Ryan Wen Liu,[1] and Jingxian Liu[1]

[1] School of Navigation, Wuhan University of Technology, Wuhan, PR China
[2] National Engineering Research Center for Water Transport Safety, Wuhan, PR China
[3] School of Engineering, Aalto University, Espoo, Finland.
*Corresponding author. Mingyang Zhang; Email: mingyang.0.zhang@aalto.fi

**Received:** 4 November 2022; **Accepted:** 25 April 2024

**Keywords:** automatic identification system; improved open window algorithm; online compression; vessel trajectory

**Abstract**

Vessel trajectories from the Automatic Identification System (AIS) play an important role in maritime traffic management, but a drawback is the huge amount of memory occupation which thus results in a low speed of data acquisition in maritime applications due to a large number of scattered data. This paper proposes a novel online vessel trajectory compression method based on the Improved Open Window (IOPW) algorithm. The proposed method compresses vessel trajectory instantly according to vessel coordinates along with a timestamp driven by the AIS data. In particular, we adopt the weighted Euclidean distance (WED), fusing the perpendicular Euclidean distance (PED) and synchronous Euclidean distance (SED) in IOPW to improve the robustness. The realistic AIS-based vessel trajectories are used to illustrate the proposed model by comparing it with five traditional trajectory compression methods. The experimental results reveal that the proposed method could effectively maintain the important trajectory features and significantly reduce the rate of distance loss during the online compression of vessel trajectories.

## Nomenclature

| | |
|---|---|
| $T$ | AIS trajectory, the data points are in the form of $\boldsymbol{t_1}, \boldsymbol{t_2}, \ldots, \boldsymbol{t_n}$. |
| $T[n]$ | The $n$th point in the set of trajectory points. |
| *Seg_TS* | A group of trajectory points compressed by the OPW algorithm, this is also used to update the trajectory point set in real-time during the compression process. |
| *anchor* | The first trajectory point of each window. |
| *floatIndex* | Slide the rightmost point in the window. |
| *isBeyond* | Boolean variable, indicating whether the calculation result is greater than the threshold. |
| *create_point*$(T[n])$ | According to the characteristics of the $n$th point in the trajectory point set $T$, a point of type *Seg_TS* is created. |
| $calculate_{PED}(T)$ | A function that returns the anchor and floating points in $T$, and a point between the anchor point and floating points, thereby producing the distance of the PED. |

| | |
|---|---|
| $calculate\_SED(T)$ | A function that returns the anchor and floating points in $T$, and a point between the anchor point and floating points, thereby returning the distance of the SED. |
| $Open\_Window\ (T, epsilon)$ | The algorithm input has two parameters: the set $T$ to be compressed and the threshold epsilon. The return value is the compressed point set $Seg\_TS$. |
| $D$ | The newly defined distance in this paper is obtained from the weighted average of the PED and SED, where the weight $\lambda$. |
| $\alpha$ | Decrease the factor chosen for the order of magnitude of $SED(T)$ |
| $i$ | Points in the slide window other than the *anchor* and *floatIndex* |

## 1. Introduction

Automatic Identification System (AIS) data mainly include three types of information: static, dynamic and semi-dynamic (Zhang et al., 2021). Static information includes details such as the name, call sign, vessel International Maritime Organisation (IMO) number, Maritime Mobil Service Identity (MMSI) and type of vessel. Dynamic information includes data such as the position (latitude and longitude), speed, course and heading of the vessel. Semi-dynamic information pertains to the ship status, cargo type, etc. (Sang et al., 2015). Thus, AIS data, especially vessel position data, are of critical importance to the fields of vessel collision avoidance, maritime supervision, follow-up investigation, vessel communication, marine accident investigation, maritime search and rescue, and comprehensive analysis of ship emissions (Ducruet, 2015; Liu et al., 2016; Wu et al., 2016; Li et al., 2017; Le Tixerant et al., 2018; Weng et al., 2020; Zhang et al., 2022, 2024).

Due to the compulsive installation of the AIS on vessels (Wei et al., 2016) and because the AIS system operates on a cyclical basis, updating and disseminating both dynamic and static information every few seconds or minutes (Xiao et al., 2017), tremendous numbers of AIS-based vessel trajectories have been collected (Zhao and Shi, 2019). However, the raw AIS dataset inherently presents challenges characterised by numerous noise points, missing data, incompleteness and redundancy. Given the critical role of data processing in feature engineering, its indispensability becomes evident in ensuring the accuracy of subsequent feature mining tasks, including denoising and compression (Huang et al., 2020). The intricacies of maritime data, often prone to discrepancies, necessitate a meticulous preprocessing approach to enhance the dataset's quality and utility. Trajectory compression can significantly simplify ship trajectories while maintaining the main geometric structure and features (Li and Yang, 2023). In the context of unsupervised route planning for Maritime Autonomous Surface Ships (MASS) at sea, Li et al. (2022) employed the adaptive Douglas–Peucker (DP) algorithm to improve the accuracy of feature extraction for MASS route planning (Douglas and Peucker, 1973; Liu et al., 2019; Zhang et al., 2023). In maritime applications, the significant influx of vessel trajectories introduces potential challenges, including expensive storage and processing costs, as well as response delays for trajectory queries. Over time, data volume may escalate to an order of magnitude beyond what computers can directly store. Achieving an optimal balance between economically storing AIS trajectories and ensuring data quality and high read speed becomes a crucial aspect of enhancing efficiency and precision in maritime operations and analysis. This balance necessitates meticulous consideration of various factors impacting the storage and accessibility of AIS trajectory data. The goal is to guarantee accuracy and timeliness for effective decision-making within the maritime domain.

According to various categorisation methods, trajectory compression can be categorised as online and offline compression. Online algorithms offer the advantage of supporting real-time applications, allowing for trajectory data compression while retrieving points from the trajectory. In contrast, offline algorithms initiate compression only after obtaining all points from the input trajectory. Generally, offline algorithms exhibit smaller errors compared with their online counterparts. Recognising the limitations of offline compression in handling dynamic and continuously expanding AIS datasets, there is a compelling need to explore and integrate the advantages offered by online compression algorithms.

https://doi.org/10.1017/S0373463324000171 Published online by Cambridge University Press

Unlike their offline counterparts, online compression algorithms have the potential to adapt in real time, accommodating the evolving nature of AIS big data and providing more effective solutions for storage and analysis. The trajectory compression can also be matched with the map for easy application (Kellaris et al., 2013). The traditional compression algorithms of DP, uniform sampling (US), spatial quality simplification heuristic (SQUISH), dead reckoning (DR) and open window (OPW) have been used for trajectory cluster analysis (Fu et al., 2005; Wang et al., 2006), trajectory classification (García et al., 2006) and trajectory data de-noising (Liu et al., 2014; Nguyen et al., 2018) of the bulk historical data with a timestamp. Compared with original historical trajectory data, compressed data can reduce a large amount of storage space, promote data processing efficiency and retain useful essential information (Zheng, 2015; Zhao and Shi, 2018; Liu et al., 2019; Qi and Ji, 2020). Additionally, some methods are improved to facilitate practical application in maritime traffic. Zhang et al. (2018) used the DP algorithm to simplify the historical vessel trajectory data from AIS for automatically planning maritime routing. Liu et al. (2019) proposed an Adaptive DP algorithm with automatic thresholding to compress the historical vessel trajectory data from AIS. While the DP algorithm is effective in preserving spatial characteristics, it falls short in handling multi-turn routes, ship speed, heading considerations and obstacle detection. To overcome these limitations, Zhou et al. (2023) introduced the multi-objective peak Douglas–Peucker (MPDP) algorithm. The algorithm optimises trajectory spatial features, heading and speed using a peak sampling strategy, and includes an obstacle-detection mechanism. It significantly advances trajectory compression, designed for enhanced performance in scenarios with curved trajectories.

The online vessel trajectory compression, which deals with the real-time reception data based on the previous compression data, is also essential for maritime traffic management. This paper focuses on an AIS data-driven dynamic compression method of vessel trajectories, which could help to obtain the near real-time updated trajectories state along with the AIS data reception. First, the traditional compression algorithms are introduced and analysed. Then, an online vessel trajectory compression approach is proposed based on the improved open window algorithm. The proposed method can help to improve maritime traffic management and collision avoidance.

## 2. Problem description and contributions

AIS relies on VHF radios to transmit data such as basic parameters, cargo information, MMSI, accurate position, course, speed, heading status and other information. Nowadays, based on using the global AIS communication link to assist the sea Internet of Things industry, carrying out intelligent traffic management and collision prevention services has become more common (Huang et al., 2020), as shown in Figure 1. Although AIS was initially developed to avoid potential hazards, it was soon used by maritime authorities to identify illegal ship activity and monitor ship behaviour. Aiming at the application of AIS in marine departments, researchers turn to the direction of trajectory data mining. The main problems that need to be solved in developing trajectory data mining techniques are the large amount of data generated worldwide and the high frequency of data emission (Wang et al., 2020). The increasing volume of data brings new challenges to the storage, transmission, processing and analysis of these data (Makris et al., 2019a, 2019b, 2021).

This paper focuses on the compression of significant ship trajectories, which can significantly reduce the computational cost required for offshore applications. Because the online compression algorithm has the advantage of supporting real-time applications, this paper proposes an improved sliding window algorithm, which compresses the ship trajectory by calculating the combined distance of perpendicular Euclidean distance (PED) and synchronous Euclidean distance (SED). Considering the spatial characteristics of the trajectory, the time attribute is introduced. However, many other data compression algorithms have not been applied to ship trajectory compression. More testing and analysis of these algorithms in ship trajectory compression are needed. Moreover, different algorithms have different characteristics, which may be highly effective in some specific data compression applications. Therefore, it is necessary to introduce the above data compression algorithms, and to study the advantages and disadvantages of these algorithms in ship trajectory compression through experiments. Therefore,

https://doi.org/10.1017/S0373463324000171 Published online by Cambridge University Press

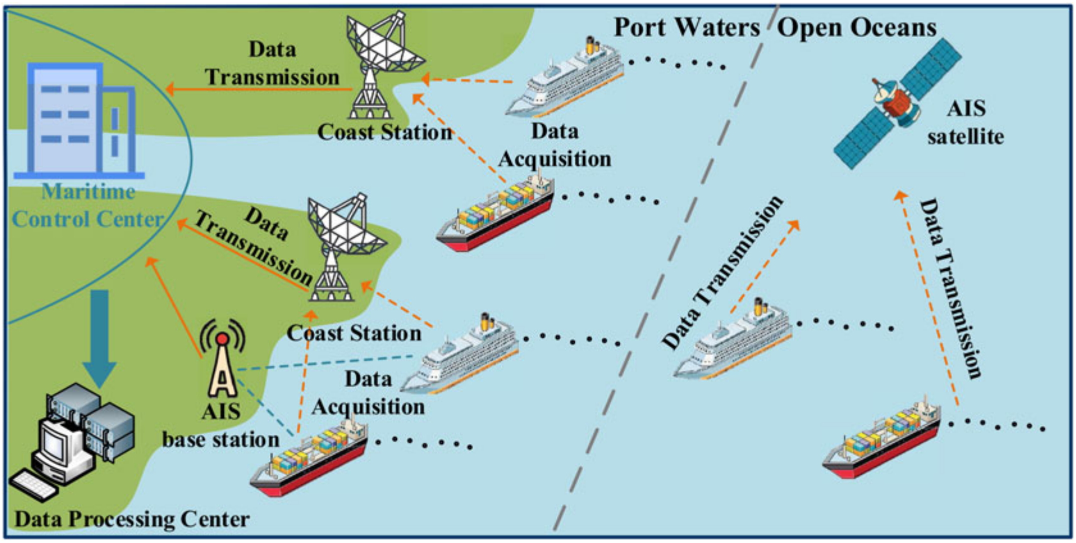


***Figure 1.*** *Terrestrial and satellite AIS communication networks.*

this paper introduces five additional classical data compression algorithms and performs experiments on compressing actual ship trajectories obtained from AIS data. The analysis of experimental results indicates that the enhanced OPW algorithm preserves more track point information during ship trajectory compression. Furthermore, the robustness of the improved OPW algorithm is validated under various conditions. The contributions of the study are as follows.

- Six trajectory compression algorithms are compared and evaluated. The algorithm is suitable for the compression of historical data or real-time data streams.
- Compression rate, distance loss rate, dynamic time warping (DTW) and running time were selected as the compression evaluation indices, considering the spatio-temporal aspects of the trajectory.
- The online and offline compression algorithms choose different thresholds according to the actual characteristics and characteristics of the trajectory to eliminate the need for any user-defined threshold.
- In the process of AIS data compression, the DP algorithm is regarded as the best-performing algorithm, followed by improved open-window and dead-reckoning. These algorithms show the most suitable performance in terms of compression rate, execution time and information loss.

The remainder of this paper is organised into several sections. Section 3 provides a literature review of the traditional trajectory compression algorithms. Section 4 introduces the AIS data pre-processing and improved OPW version and the performance indices for evaluating trajectory compression. Section 5 briefly introduces the sources of AIS-based ship trajectories. Furthermore, in this section, extensive experiments are performed on several different trajectory compression algorithms, and the experimental results are analysed. In conclusion, this paper encapsulates its primary contributions and outlines prospective avenues for future research in Section 6.

## 3. Traditional compression algorithms

The AIS data-based vessel trajectory is a set of points and can be represented as $P = (p_1, p_2, \cdots, p_i, \cdots, p_n)$. Here, $p_i$ is the $i$th vessel position in time series and $n$ expresses the total number of points. The trajectory can be obtained by creating lines that connect the points in sequence. The goal of vessel trajectory compression is to obtain a new set of $P' = ({p'}_1, {p'}_2, \cdots, {p'}_i, \cdots, {p'}_m)$ by retaining characteristic points and clearing redundant points. The lines connecting the points of $P$ in sequence approximate the lines connecting the points connecting the points of $P'$ in sequence.

https://doi.org/10.1017/S0373463324000171 Published online by Cambridge University Press

The four algorithms of DP, US, SQUISH and DR are traditional offline compression algorithms, which are widely adopted in the batched compression of historical trajectory data. Meanwhile, the OPW algorithm is a traditional online compression algorithm, which is adopted in road traffic.

### *3.1. Douglas–Peucker algorithm*

The Douglas–Peucker algorithm, also called the DP algorithm, was proposed for simplifying two-dimensional curves (Douglas and Peucker, 1973). The DP algorithm has also been used to compress vessel trajectory data (Liu et al., 2019, 2020). A schematic diagram of the DP algorithm is shown in Figure 2. The vessel trajectory compression procedure based on the DP algorithm is as follows.

- **Step 1.** Set a threshold according the acceptable margin of error for practical application. Then, the first point $p_1$ and last point $p_n$ of the trajectory are respectively selected as the initial anchor point and initial floating point to create a baseline $\overline{p_1 p_n}$. Next, the distances of the other points to the baseline are calculated one by one to find the point with the maximum distance. If the maximum distance is smaller than the setting threshold, the points except the first point $p_1$ and last point $p_n$ should be cleared, which means that the first point $p_1$ and last point $p_n$are the characteristic points and the compressed trajectory data are the set of $P' = (p_1, p_n)$. Otherwise, the next step is taken.
- **Step 2.** The point with the maximum distance greater than the setting threshold is a reference characteristic point, acting as both the floating point corresponding to the previous anchor point and the anchor point corresponding to the previous floating point in Step 1. Suppose the reference characteristic point is $p_k$. The $\overline{p_1 p_k}$ is the new baseline of points of $(p_1, p_2, \cdots, p_i, \cdots, p_k)$, and the $\overline{p_k p_n}$ is the new baseline of points of $(p_k, p_{k+1}, \cdots, p_j, \cdots, p_n)$. Then, the distances of the points of $(p_2, \cdots, p_i, \cdots, p_{k-1})$ to the baseline of $\overline{p_1 p_k}$ are calculated one by one to find the point with the maximum distance, and the existence of a new reference characteristic point in $(p_1, p_2, \cdots, p_i, \cdots, p_k)$ is checked by comparing the value of the maximum distance and the setting threshold. Additionally, the points of $(p_k, p_{k+1}, \cdots, p_j, \cdots, p_n)$ and the baseline of $\overline{p_k p_n}$ are also handled for checking the existence of a new reference characteristic point. If there are no new reference characteristic points, the compressed trajectory data are the set of $P' = (p_1, p_k, p_n)$. Otherwise, the next step is taken.
- **Step 3.** The points corresponding to the baseline with the existing new reference characteristic points continue to be processed like Step 2 until there is no unique reference characteristic point in each subset of points. The first and last point as well as all reference characteristic points are retained as the compressed trajectory data.

The threshold is the basis of the DP algorithm. A more significant threshold may cause that compressed trajectory to have low similarity with the original trajectory. Meanwhile, a smaller threshold may cause a low compression ratio to result in failure of achieving a satisfactory outcome. In addition, the DP algorithm is time-consuming due to a large amount of distance calculation.

### *3.2. Uniform sampling algorithm*

The US algorithm is used to compresses the trajectory data by uniformly selecting the sample data in the time sequence. Suppose the interval of retained trajectory samples is $m$, then the compressed data of trajectory $P$ is $(p_1, p_{m+1}, \cdots, p_{im+1}, \cdots)$. However, the US algorithm can't effectively compress the trajectory data with large fluctuation.

### *3.3. Spatial quality simplification heuristic algorithm*

The SQUISH algorithm is proposed for offset data compression by deleting the points that cause an acceptable margin of error (van Emde Boas, 1975; Muckell et al., 2011, 2014). The schematic diagram of the SQUISH algorithm is shown in Figure 3. The effect of data compression depends on the size

https://doi.org/10.1017/S0373463324000171 Published online by Cambridge University Press

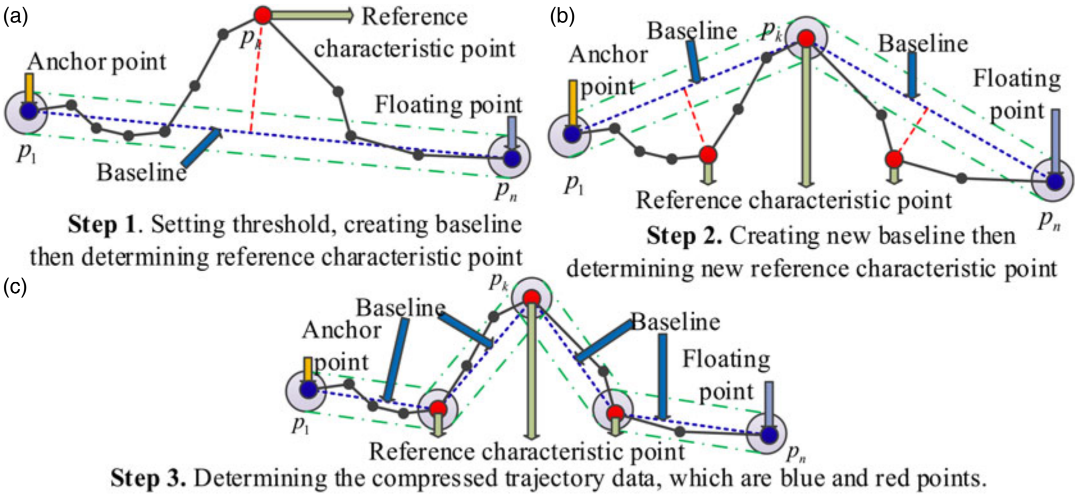


***Figure 2.*** *Schematic diagram of DP algorithm.*

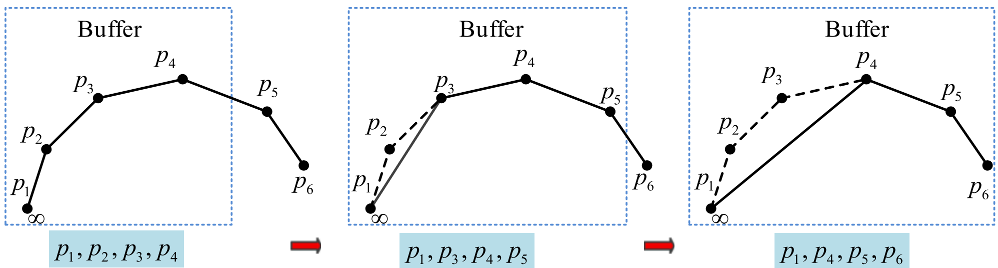


***Figure 3.*** *Schematic diagram of SQUISH algorithm.*

of buffer based on the SQUISH algorithm. The vessel trajectory compression procedure based on the SQUISH algorithm is as follows.

- **Step 1.** Set a buffer size, which determines the number of the trajectory points after compression according to the compression ratio. Suppose there are six points of $(p_1, p_2, \cdots, p_i, \cdots, p_6)$ in the original vessel trajectory dataset and four points in the buffer.
- **Step 2.** Input a new point of vessel trajectory into the buffer. When the next point is recorded in the buffer, the point among the previous points with the smallest error will be deleted. Then, a new set in the buffer will be obtained. Continually, input the points by the timestamp, which are not in the buffer, and delete the points with the smallest error until there are no points out of the buffer.
- **Step 3.** Finally, the complete compressed vessel trajectory data with a set of $m$ points will be obtained. In Figure 2, the points of $(p_2, p_3)$ have the smallest error in turn, so the points of $(p_1, p_4, p_5, p_6)$ are the compression trajectory.

### *3.4. Dead reckoning algorithm*

The DR algorithm is a compression algorithm that uses the integral vector technique of estimating or measuring displacement to the fixed position (Zhao et al., 2003; Trajcevski et al., 2006). The schematic diagram of the DR algorithm is shown in Figure 4. Predicting ship position from ship speed and previous position is the basis for ship trajectory compression based on the DR algorithm. The vessel trajectory compression procedure based on the DR algorithm is as follows.

https://doi.org/10.1017/S0373463324000171 Published online by Cambridge University Press

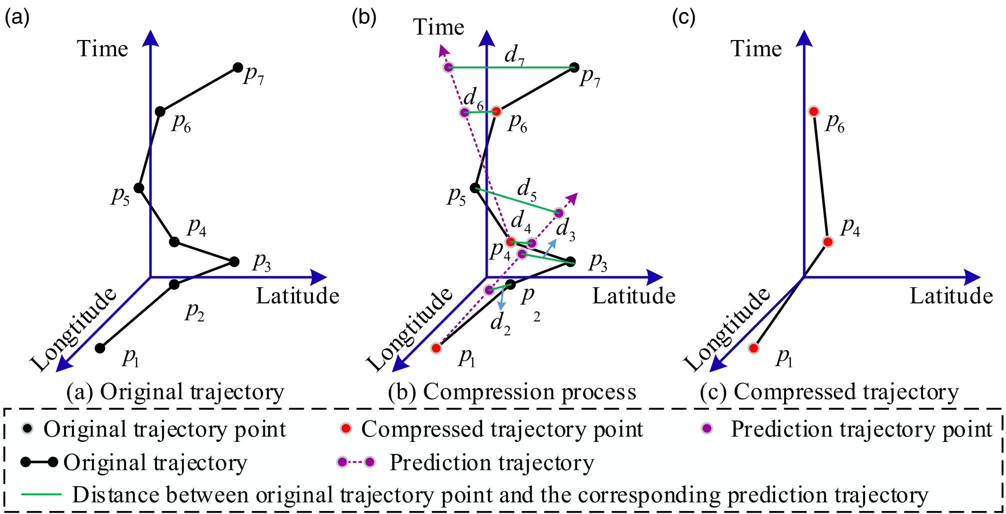


***Figure 4.*** *Schematic diagram of DR algorithm.*

- **Step 1.** Set an error threshold, which determines the retained points of vessel trajectory data. Suppose there are seven points of $(p_1, p_2, \cdots, p_i, \cdots, p_7)$ in the original vessel trajectory dataset and the error threshold is $d_0$.
- **Step 2.** The ship's trajectory is predicted based on the first point, and the speed and the distance between the original and the corresponding trajectory points are calculated in the time series. Furthermore, the previous point of the point with a distance more significant than the error threshold is a compressed trajectory point, which is the subsequent base point to predict the vessel trajectory. This process is constantly implemented until the last point of the compressed trajectory is determined.
- **Step 3.** Determine the compression trajectory data. In Figure 3, $d_5 > d_0$ and $d_7 > d_0$, so the points of $(p_1, p_4, p_6)$ are the compression trajectory.

### 3.5. *OPW algorithm*

The OPW algorithm is a classical online compression algorithm (Keogh et al., 2001), which compresses the trajectory data based on the window line of the connection between the two points. The vessel trajectory compression procedure based on the OPW algorithm is as follows.

- **Step 1.** Set a threshold, which determines the retained points of vessel trajectory data. Suppose there are $n$ points of $(p_1, p_2, \cdots, p_i, \cdots, p_n)$ in the original vessel trajectory dataset.
- **Step 2.** Determine a window line by connecting the first point $p_1$ and the third point $p_3$ in the time series, and calculate the distance from the second point $p_2$ to the window line. If the distance is smaller than the threshold, the second point should not be retained in the compressed dataset. Otherwise, the second point should be retained in the compressed dataset. Next, a new window line should be replaced as the connection between the second point and the fourth point, and afterwards, the distance from the third point to the new window line is calculated to determine whether the third point should be in the compressed dataset or not. The compression will continue until the last point of the trajectory.
- **Step 3.** Determine the compression trajectory data. Meanwhile, there is some improved OPW algorithm used for compression trajectory. The before open window (BOPW) algorithm is used to compress the trajectory data by changing the window line, and there are two or more points between the former point and the latter point. Moreover, when the sum of errors in the window is greater than a threshold, the second-to-last trajectory point in the window is used as the end point of the

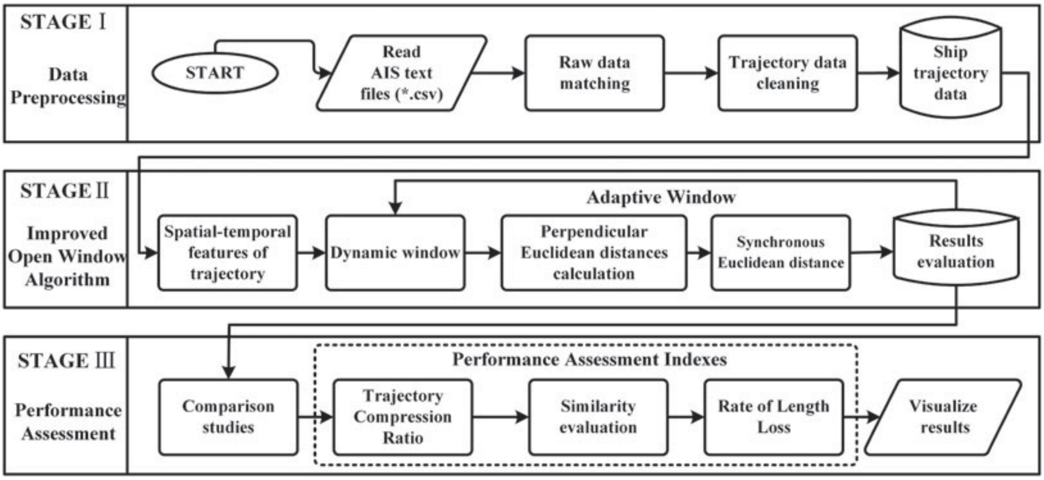


***Figure 5.*** *Logic framework of ship trajectories compression.*

approximate line segment of that window. Another improved OPW algorithm is called the normal open window (NOPW) algorithm, which changes the window line like BOPW, but it generates a data point in the window that exceeds the error threshold as the end point of the approximate line segment of the window (Muckell et al., 2010). In addition, there is an improved OPW algorithm that uses SED to replace PED (Meratnia and Rolf, 2004).

## 4. Methodology

The paper proposes an online ship trajectory compression method for massive AIS data storage. The proposed method considers the spatial temporal features of ship trajectory and comprises the three steps summarised in Figure 5.

### *4.1. Step 1: AIS vessel trajectory data preprocessing*

The information from AIS consists of the MMSI number, timestamp, position, speed, course and other vessel attributes (Wang et al., 2022). However, the vessel trajectory data always have abnormal data, due to the equipment. The abnormal data, which is caused by data loss and other phenomena, will be deleted to improve the dataset quality by data preprocessing (Zhao et al., 2018a, 2018b).

For the vessel AIS information in the relevant area, the vessel position information collected by the AIS equipment at times $T_1$ and $T_2$ are $(\lambda_1, \varphi_1)$ and $(\lambda_2, \varphi_2)$. If $T_1 < T < T_2$, then the position information of vessel AIS at time $T$ can be calculated by the following formulae:

$$\lambda_T = \lambda_1 \times \frac{T_2 - T}{T_2 - T_1} + \lambda_2 \times \frac{T - T_1}{T_2 - T_1} \tag{1}$$

$$\varphi_T = \varphi_1 \times \frac{T_2 - T}{T_2 - T_1} + \varphi_2 \times \frac{T - T_1}{T_2 - T_1} \tag{2}$$

For large and medium-sized vessels, their course and speed will not be significantly changed in a short time, nor will their bow direction change significantly, which can be approximated to uniform linear motion. However, for small vessels, their manoeuvrability is better. When the speed and course change rate is significant, some errors will be caused. At this point, the vessel classification as a small vessel is derived from the size data obtained through the AIS. Subsequently, the rate of change of the vessel's speed and course is computed, leading to a refinement of the previously presented formulae.

https://doi.org/10.1017/S0373463324000171 Published online by Cambridge University Press

### *4.2. Step 2: AIS vessel trajectory compression based on improved open window algorithm*

An improved algorithm to compress the AIS trajectory data generated by inland and coastal vessels is proposed to explore whether online compression technology can better compress vessel trajectory data. The algorithm based on an existing online compression technology called the OPW (Keogh et al., 2001) is an IOPW algorithm. In addition, various performance indicators are used in this context. The OPW algorithm first sets a window for data compression, containing some trajectory points. Unlike other window algorithms, however, the OPW algorithm then calculates the PED for all the points in the window. Based on the relative error between the sum of the errors within the window and a given threshold, the endpoints of the approximate line segments can be selected.

The OPW algorithm relies on the distance generated by the combination of PED. Although PED can retain the contour information of the vessel trajectory, its time information is almost random. SED is introduced to limit the trajectory's offset further to achieve better information retention. Compared with the OPW algorithm, which only considers the spatial features of the trajectory points, the IOPW algorithm also finds the time characteristics of the trajectory. This algorithm employs iterative methods to measure the amount of information loss incurred by trajectory simplification, but no longer iterates over the entire trajectory. The OPW algorithm uses the established window to iterate and continuously update the window information until simplification is completed.

The improved OPW algorithm is illustrated in Figure 6. The original OPW algorithm uses the PED as a measure to calculate the error, for example, $|\overrightarrow{\mathbf{t}_4\mathbf{t}_4^*}|$ is the PED of data point $\mathbf{t}_4$ to line segment $\overrightarrow{\mathbf{t}_0\mathbf{t}_{10}}$ in Figure 7(a). The trajectory data compression algorithm using the PED has a good compression ratio (Cao et al., 2006; Shi and Cheung, 2006; Liu et al., 2015). However, the time information is ignored in the PED. Therefore, when the PED line segment simplification algorithm is applied in the spatio-temporal query of a compressed trajectory to determine 'the position of the object at time $m$', and it returns an approximate perpendicular point $\mathbf{t}_m^*$. However, the distance from that point to the actual position $\mathbf{t}$ of the moving object at time $m$ is not limited. In the present study, the coordinates of the $\mathrm{t}_i$,$\mathrm{t}_j$ and $\mathbf{t}_m$ $(\boldsymbol{i} < \boldsymbol{m} < \boldsymbol{j})$ trajectory points in the original trajectory sequence are $(x_i,\ y_i,\ z_i)$, $(x_j, y_j, z_j)$, $(x_m, y_m, z_m)$ (the $z$ coordinates represent time information). Thus, the PED of data point $\mathbf{t}_m$ to point $\mathbf{t}_m^*$ (assuming its coordinates are $(x_m^*, y_m^*, z_m^*)$ and the window line formed by $\mathbf{t}_i$,$\mathbf{t}_j$) can be obtained as

$$\mathrm{PED}(\mathbf{t}_m) = \left| \frac{(y_j - y_i)x_m + (x_i - x_j)y_m + x_j y_i - x_i y_j}{\sqrt{(y_j - y_i)^2 + (x_j - x_i)^2}} \right| \tag{3}$$

Figures 7(a) and 7(b) show the trajectory $T[\mathbf{t}_0, \ldots, \mathbf{t}_{10}]$ with a total of 11 points, represented by two and four continuous segments (black) and compressed by the DP algorithm using PED and SED, respectively. In the two figures, the DP algorithm first creates a line segment $\overrightarrow{\mathbf{t}_0\mathbf{t}_{10}}$. Then, it calculates the distance from the other points on the trajectory to the $\overrightarrow{\mathbf{t}_0\mathbf{t}_{10}}$ in panel (a). The DP algorithm finds that point $\mathbf{t}_4$ has the maximum distance to $\overrightarrow{\mathbf{t}_0\mathbf{t}_{10}}$, being greater than the specified error threshold. Therefore, the trajectory is divided into two segments $[\mathbf{t}_0, \ldots, \mathbf{t}_4]$ and $[\mathbf{t}_4, \ldots, \mathbf{t}_{10}]$.

The SED is the Euclidean distance between the data points and the approximate spatio-temporal data points on the corresponding line segment (van Emde Boas, 1975). The approximate spatio-temporal data point $\mathbf{t}'$ is obtained according to the point $\mathbf{t}$ on the original trajectory ($\mathbf{t}'$ can be obtained from the first and last points of the trajectory according to interpolation in time), and just calculates the distance between $\mathbf{t}'$ and $\mathbf{t}$. After the original text is modified, SED should be calculated by trajectory points and their approximate spatio-temporal data points. In Figure 7(b), $\mathbf{t}'_2$,$\mathbf{t}'_4$ and $\mathbf{t}'_7$ are approximate spatio-temporal data points of $\mathbf{t}_2$, $\mathbf{t}_4$, and $\mathbf{t}_7$, and more trajectory points can be preserved using the SED compression algorithm. Meanwhile, the SED can be used to limit the error threshold between trajectory points and their approximate spatio-temporal data points (e.g. $\mathbf{t}'_4$ is the point where $\mathbf{t}_0$ and $\mathbf{t}_{10}$ interpolate in proportion to time). The SED represents the Euclidean distance between the point on the original path

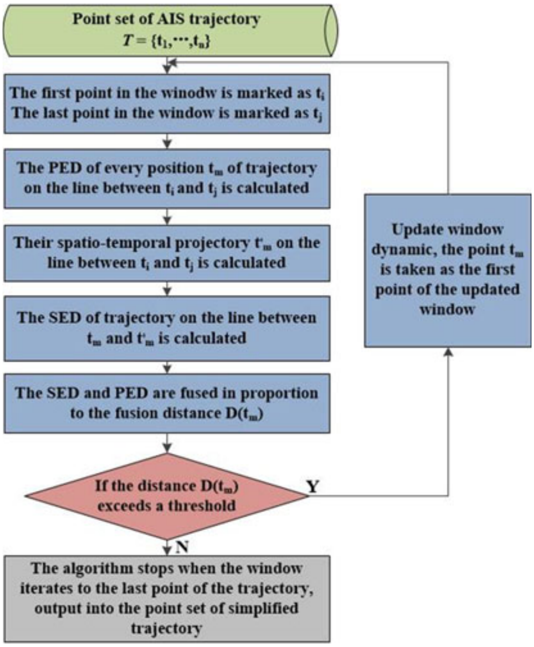


***Figure 6.*** *Flow chart of IOPW algorithm.*

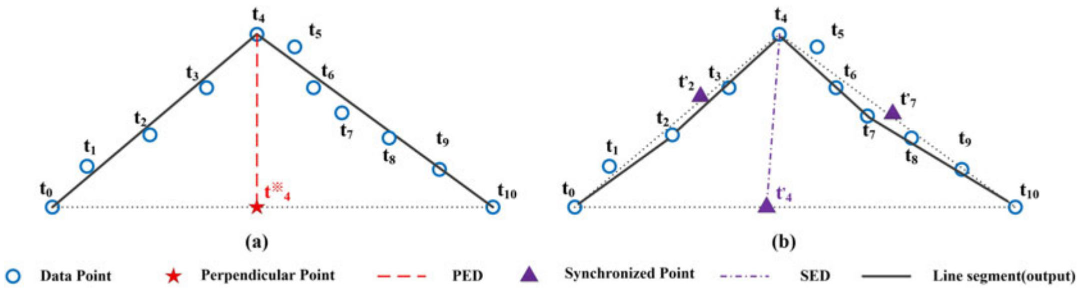


***Figure 7.*** *(a) PED and (b) SED illustrate by using* $T[\boldsymbol{t}_0, \ldots, \boldsymbol{t}_{10}]$ *and* $T'[\boldsymbol{t}_0, \boldsymbol{t}_4, \boldsymbol{t}_{10}]$.

and the point on the simplified path in proportion to time in a general way. Similarly, if it is assumed that the trajectories from $\mathbf{t}_i$ to $\mathbf{t}_j$ in the original trajectory sequence are compressed into $\mathbf{t}_i$ and $\mathbf{t}_j$, then $\mathbf{t}_\mathbf{m}$ is located in the middle of $\mathbf{t}_i$ and $\mathbf{t}_j$ assuming the coordinates are $(x_m,\ y_m,\ z_m)$. The time scale projection position $\mathbf{t'}_m$ on the trajectory segment is$(x'_m,\ y'_m,\ z'_m)$:

$$x'_m = x_i + \frac{z_m - z_i}{z_j - z_i}(x_j - x_i),\ z_i < z_m < z_j \tag{4}$$

$$y'_m = x_i + \frac{z_m - z_i}{z_j - z_i}(y_j - y_i),\ z_i < z_m < z_j \tag{5}$$

When the position of $\mathbf{t'}_m$ is computed according to $\mathbf{t}_i$ and $\mathbf{t}_j$ by interpolation in time, the SED distance of $\mathbf{t}_m$ to $\mathbf{t'}_m$ can be calculated:

$$\mathrm{SED}(\mathbf{t}_m) = \sqrt{(x_m - x'_m)^2 + (y_m - y'_m)^2} \tag{6}$$

Therefore, with the IOPW algorithm using the fusion distance between SED and PED as a parameter to compare with the error threshold, $\mathrm{D}(\mathbf{t}_\mathbf{m})$ is the newly defined fusion distance, based on the original

https://doi.org/10.1017/S0373463324000171 Published online by Cambridge University Press

algorithm using the PED distance as the measurement error threshold. Since there is an order of magnitude difference between SED and PED, $\alpha$ is multiplied by the formula to unify the order of magnitude of SED and PED. The new parameter SED is continuously introduced. Combining the two distances and the resulting distance D, the value of D($\mathbf{t_m}$) is determined through the application of Equations (3) and (6):

$$D(\boldsymbol{t_m}) = \lambda \times \mathrm{PED}(\boldsymbol{t_m}) + (1 - \lambda) \times \alpha \times \mathrm{SED}(\boldsymbol{t_m}) \tag{7}$$

After multiple experiments, it was found that when $\alpha = 0.01$, $\lambda = 0.87$, better compression results can be obtained. As described in Section 5, the above parameters were used for the experiments.

In the present study, the IOPW algorithm is employed with the OPW method to determine the end point of the approximate linear segment. The trajectory of a vessel can be represented by a set of points $T = \{\mathbf{t_i}|\mathbf{t_i}, \mathbf{i} = 1, 2, 3, 4 \ldots, n\}$, where $\mathbf{t_i}$ represents an AIS information point on the vessel trajectory and $n$ is the number of points. Window lines were constructed from $\mathbf{t}_1$ to $\mathbf{t_i}$ ($\mathbf{i} \geq 3$). The PED(T) is first calculated from the point between $\mathbf{t_1}$ and $\mathbf{t_i}$ to the window line. Second, the synchronisation point $\mathbf{t'_i}$ of the $\mathbf{t_i}$ point within the error range is determined, and the SED from $\mathbf{t_i}$ to $\mathbf{t'_i}$ is calculated. Here, $\mathbf{D}$ ($\mathbf{t_i}$) is calculated by the relation D($\mathbf{t_m}$) = $\lambda \times$PED($\mathbf{t_m}$) + $(1 - \lambda) \times \alpha \times$ SED($\boldsymbol{t_m}$). If $\mathbf{D}$ ($\mathbf{t_i}$) > epsilon, $\mathbf{t_i}$ is regarded as the end point of the approximate trajectory of the previous segment and is also used as the starting point for the subsequent compression simplification. This point is also saved as a compressed point to the point set $F = \{f(k)|k = 1, 2 \ldots, m\}$, where $k$ is the number corresponding to the specific point and $m$ is the number of points. If the point between $\mathbf{t}_1$ and $\mathbf{t_i}$ does not have a vertical Euclidean distance exceeding the threshold, the point $\mathbf{t_{i+1}}$ after $\mathbf{t_i}$ is introduced to the window. The operation process is then repeated. The symbols and terminology used in the algorithm pseudo code and the pseudo code itself are given in the Nomenclature section and Table 1, respectively.

Figure 7 shows that the AIS data points exceeding the threshold are segmentation points. The data sequence is interrupted at data points $\mathbf{t_4}$, $\mathbf{t_8}$, $\mathbf{t_{12}}$ and $\mathbf{t_{16}}$. There are multiple termination conditions for each iteration of the window algorithm, e.g. based on the number of points, the SED and PED, or the maximum error in the window. Once the threshold for any of these conditions is not satisfied, the current iteration is terminated and the next iteration is initiated. The fusion distance formed by SED and PED at $\mathbf{t_4}$ point exceeds the error threshold. So $\mathbf{t_4}$ is retained in the compressed trajectory. Meanwhile, the previous iteration ends and $\mathbf{t_4}$ is taken as the initial trajectory point of the new iteration to continue the compression. Finally, the points retained on the compressed trajectory are $\mathbf{t_1}$, $\mathbf{t_4}$, $\mathbf{t_8}$, $\mathbf{t_{12}}$ and $\mathbf{t_{16}}$, and after the trajectory point $\mathbf{t_{19}}$, the compression will continue until the final point of the trajectory. Therefore, countermeasures to address this problem are required. As the algorithm only considers the position of the AIS trajectory point in the window, it can achieve the best performance in each provincial simplification. However, it cannot balance the overall trend.

Although the computational cost of the OPW algorithm is high, its application in road traffic trajectory compression is widespread (Cao et al., 2010). Most of this is attributable to the fact that the algorithm is less affected by noise when computing relatively short data sequences. The IOPW algorithm can perform dynamic data calculation. As the input, only the start point of the trajectory is required. The algorithm does n't need to determine the trajectory endpoint.

### *4.3. Step 3: Performance assessment indices*

The performance indices examined in the compression experiments are delineated below.

#### *4.3.1. Trajectory compression ratio*

The compression ratio measures how much storage capacity has been saved by compressing original vessel trajectories (Muckell et al., 2011). The higher compression ratio is related to effectively reducing the storage cost while preserving the utility of trajectories. The trajectory compression ratio (TCR)

***Table 1.*** *IOPW algorithm.*

**Algorithm 1.** IOPW compression considering course

```
INPUT:
T={t1, t2, . . . , tn}        //original trajectory
Epsilon                       // error tolerance
OUTPUT:
Seg_TS                        //compressed trajectory
i = 1
anchor = 2
floatIndex = 3
is Beyond = true
Seg_TS = Concat (Seg_TS, create_point(T[anchor])
While(floatIndex≤T.size())
{
   If(is Beyond = = true){
      D = (calculate_PED (T[anchor], T[i+1], T[floatIndex]) * λ +calculate_SED (T[anchor],
T[i+1], T[floatIndex]) * (1-λ)*α)
   }
   Else{
      D = (calculate_PED (T[anchor], T[i], T[floatIndex]) * λ +calculate_SED (T[anchor], T[i], T[floatIndex]) * (1-λ)*α)
   }
   If (D> epsilon){
      is_Beyond = true
      i = i+1
      anchor = i
      Seg_TS = Concat (Seg_TS, create_point(T[anchor])
      floatIndex = anchor + 2
   }
   Else{
      is_Beyond = false
      floatIndex = floatIndex + 1
      anchor = anchor + 1
   }
}
Return Seg_TS
```

adopted in this study is mathematically defined as follows:

$$\mathrm{TCR} = \left(1 - \frac{N_c}{N_o}\right) \times 100\% \tag{8}$$

where $N_c$ and $N_o$ denote the numbers of timestamped points in compressed and original vessel trajectories, respectively.

*4.3.2. Dynamic time warping*

Upon achieving satisfactory compression results, the theoretical standpoint suggests a high degree of similarity between the original and compressed trajectories of the vessel. It is commonly tricky to

https://doi.org/10.1017/S0373463324000171 Published online by Cambridge University Press

accurately measure the similarities since the numbers of timestamped points differ in the original and compressed versions. Dynamic time warping (DTW) will be adopted to effectively evaluate the compression quality to calculate the distances (inversely proportional to the similarities) between different vessel trajectories. The basic principle behind DTW is to use the dynamic programming method to find the minimum distance between every two trajectories. In the conducted experiments, small distances (indicative of high similarities) are inherently associated with high-quality trajectory compression. Please refer to Keogh and Ratanamahatana (2005) for more details on DTW.

#### *4.3.3. Rate of length loss*

The rate of length loss (RLL) essentially indicates the rate of the length loss and the total length of one original vessel trajectory (Huang et al., 2020). Theoretically, the low value of RLL is naturally related to high-quality trajectory compression, which could preserve the main geometrical features in the compressed trajectories. In this study, the variables $\{S_1, S_2, \cdots, S_M\}$ and $\{S'_1, S'_2, \cdots, S'_M\}$ are designated respectively denoting the original and compressed vessel trajectories. Here, the symbol $M$ denotes the total count of vessel trajectories employed in the experiments. The mathematical definition of RLL is given by

$$\text{RLL} = \frac{\sum_{m=1}^{M} |S_m| - \sum_{m=1}^{M} |S'_m|}{\sum_{m=1}^{M} |S_m|} \tag{9}$$

where $\sum_{m=1}^{M} |S_m| - \sum_{m=1}^{M} |S'_m|$ is the length loss after trajectory compression, $|S_m|$ and $|S'_m|$ denote the lengths of the $m$th original trajectory and its compressed version. In particular, the trajectory length can be regarded as the sum of the geometrical distance between any two adjacent timestamped points in one trajectory.

#### *4.3.4. Running time*

The running time represents the execution time of CPU implementation for the trajectory compression task. It will be adopted to evaluate the efficiency of the compression algorithms for three different water areas. The experiments exclusively address the runtime involved in compressing vessel trajectories, excluding the computational expenses associated with data loading and transfer.

#### *4.3.5. Contrastive analysis*

The compression algorithm consists of an offline compression algorithm (also called an offline algorithm), e.g. DP algorithm and US algorithm, and an online compression algorithm, e.g. SQUISH algorithm, DR algorithm, OPW algorithm and IOPW algorithm. The DP algorithm is a popular batch-processing algorithm for line segment simplification. The US algorithm is a highly efficient compression method for batch data. The online compression algorithm is more efficient and conducive to real-time maritime monitoring than offline compression. Additionally, the time complexity of an offline compression algorithm is generally higher than that of an online compression algorithm. The SQUISH algorithm is an online compression algorithm with superior performance that is achieved by simplifying the original trajectory using local optimisation. The OPW algorithm is an online compression algorithm that can perform local optimisation according to window compression processing. The DR algorithm cannot estimate the cumulative error and offset for online trajectory compression of motion state changes. The details of the traditional compression algorithms are listed in Table 2.

## 5. Experimental results and discussion

To evaluate and compare the competing compression methods, a set of experiments on realistic AIS-based vessel trajectories will be performed in terms of both quantitative and qualitative evaluations. All experiments will be implemented using C++ with an Integrated Development Environment (IDE) on a machine with a 2.8-GHz Intel Core i7-7700HQ CPU and 8 GB of RAM. The proposed IOPW

https://doi.org/10.1017/S0373463324000171 Published online by Cambridge University Press

***Table 2.*** *Details of the traditional compression algorithms.*

| Arithmetic types | Algorithms | Details | |
|---|---|---|---|
| | | Computational complexity[a] | Error criteria |
| Batch | DP | $O(N), O(N^2), O(N \times \log N)$ | SED distance |
| | US | $O(N)$ | Compression rate |
| Online | SQUISH | $O(N \times \log m)$ | SED distance |
| | DR | $O(N)$ | SED distance, velocity |
| | OPW | $O(N^2)$ | SED distance |
| | IOPW | $O(N^2)$ | SED distancePED distance |

a*m* indicates the queue size and *N* indicates the number of variables.

***Table 3.*** *Statistical information related to three different water areas.*

| Water areas | Period | Number of vessel trajectories | Number of timestamped points |
|---|---|---|---|
| MarineCadastre | 2016-04-09 | 596 | 87, 322 |
| Wuhan Section of the Yangtze River (WSYR) | 2016-08-01 | 401 | 62, 582 |
| South Channel of the Yangtze River Estuary (SCYRE) | 2017-08-09 | 1, 476 | 1, 008, 270 |

will be compared with five competing methods, i.e. DP, uniform sampling, SQUISH, DR and OPW. The realistic AIS-based vessel trajectories employed in the experiments are curated from three distinct water areas, i.e. the MarineCadastre, the Wuhan Section of the Yangtze River (WSYR) and the South Channel of the Yangtze River Estuary (SCYRE). Meanwhile, four performance indices, i.e. trajectory compression rate (TCR), trajectory similarity using DTW, distance loss rate using RLL, and running time are simultaneously adopted to evaluate the compression results quantitatively.

### *5.1. Data sources*

The MMSI number and time stamp were used to obtain AIS data for all the vessels in a specific area. For the three water areas considered herein, i.e. MarineCadastre, WSYR and SCYRE, 596, 401 and 1,476 vessel trajectories that could represent traffic patterns were chosen to form the experiment data sources. The details of the dataset are given in Table 3.

### *5.2. Experimental results*

In this study, $0 \cdot 001$, $0 \cdot 003$, $0 \cdot 005$ and $0 \cdot 04$ were selected as the algorithm error threshold, and these step sizes were applied in the DP, DR, OPW and uniform sampling algorithms. For the SQUISH algorithm, the compression rate was taken as the algorithm reciprocal of the compression ratio with values of $0 \cdot 01$, $0 \cdot 04$, $0 \cdot 07$ and $0 \cdot 20$. Parameters of 3, 7, 10 and 60 were selected for the uniform sampling algorithm. To verify the advantages of IOPW, parallel experiments were performed involving nine different parameters for each of the five algorithms. The experimental results are listed in Tables 4–6.

Table 4 displays the compression rate, DTW matching distance, RLL and running time on the application of the algorithms to the AIS data of vessels on the Wuhan Section of the Yangtze River.

https://doi.org/10.1017/S0373463324000171 Published online by Cambridge University Press

***Table 4.*** *Comparative studies for the Wuhan section of the Yangtze river.*

| | Algorithms | Thresholds | Number | TCR | DTW | RLL | Time (s) |
|---|---|---|---|---|---|---|---|
| Batch | DP | 0·001 | 5,967 | 6·80% | $5\cdot328\times10^{-3}$ | 0·366% | 1·496 |
| | | 0·003 | 3,404 | 3·88% | $9\cdot379\times10^{-3}$ | 1·295% | 1·029 |
| | | 0·005 | 2,505 | 2·86% | $1\cdot647\times10^{-2}$ | 2·499% | 0·816 |
| | | 0·040 | 1,168 | 1·33% | $3\cdot539\times10^{-2}$ | 6·926% | 0·423 |
| | US | 3 | 29,638 | 33·79% | $1\cdot567\times10^{-2}$ | 14·353% | 0·112 |
| | | 7 | 13,051 | 14·88% | $3\cdot303\times10^{-2}$ | 17·881% | 0·114 |
| | | 10 | 9,311 | 10·61% | $3\cdot223\times10^{-2}$ | 22·878% | 0·062 |
| | | 60 | 2,053 | 2·34% | $6\cdot550\times10^{-2}$ | 26·673% | 0·054 |
| Online | SQUISH | 0·01 | 1,047 | 1·19% | $9\cdot088\times10^{-2}$ | 19·958% | 1·914 |
| | | 0·04 | 3,408 | 3·89% | $4\cdot746\times10^{-2}$ | 5·027% | 7·016 |
| | | 0·07 | 5,978 | 6·81% | $3\cdot556\times10^{-2}$ | 2·866% | 11·162 |
| | | 0·20 | 17,394 | 19·83% | $1\cdot647\times10^{-3}$ | 0·559% | 25·658 |
| | DR | 0·001 | 9,915 | 11·30% | $2\cdot405\times10^{-2}$ | 13·168% | 0·990 |
| | | 0·003 | 5,522 | 6·29% | $5\cdot235\times10^{-2}$ | 21·864% | 1·021 |
| | | 0·005 | 4,210 | 4·80% | $3\cdot957\times10^{-2}$ | 23·108% | 1·057 |
| | | 0·040 | 1,575 | 1·80% | $7\cdot175\times10^{-2}$ | 26·048% | 0·998 |
| | OPW | 0·001 | 15,796 | 18·01% | $2\cdot113\times10^{-3}$ | 0·161% | 8·906 |
| | | 0·003 | 8,156 | 9·30% | $5\cdot442\times10^{-3}$ | 0·464% | 9·298 |
| | | 0·005 | 6,623 | 7·55% | $7\cdot861\times10^{-3}$ | 0·724% | 14·518 |
| | | 0·040 | 3,923 | 4·47% | $2\cdot492\times10^{-2}$ | 5·030% | 35·101 |
| | IOPW | 0·001 | 11,893 | 13·56% | $2\cdot993\times10^{-3}$ | 0·205% | 8·512 |
| | | 0·003 | 6,895 | 7·86% | $6\cdot448\times10^{-3}$ | 0·556% | 20·006 |
| | | 0·005 | 5,791 | 6·60% | $8\cdot822\times10^{-3}$ | 0·907% | 29·383 |
| | | 0·040 | 3,157 | 3·60% | $2\cdot603\times10^{-2}$ | 5·867% | 60·484 |

The results demonstrate that the compression rates of the DR, uniform sampling, DP, OPW and IOPW increase with the parameter values. The DP algorithm has a higher running rate than IOPW, whereas the compression rate of the two algorithms is very close when the threshold is significant. However, the average DTW matching distance and RLL of the IOPW algorithm are generally smaller than those of the traditional DP algorithm at the same threshold. The compression performance of the DR algorithm is closer to that of IOPW. However, the information retention of the compressed AIS trajectory is lower than that of IOPW. The SQUISH algorithm has the best preservation effect on the trajectory information when the compression rate is 19·83%. When the compression rate is significant, the SQUISH compression performance is poor, but the trajectory information retention is better.

Further, when the parameter is changed from 0·01 to 0·04, the RLL is changed from 19·958% to 5·027%. Generally, the AIS data transmission frequency is within 2–120 s. Therefore, the AIS emission frequency and terrain are considered to be the causes of large fluctuations in indicators such as distance loss. Little change was observed for other datasets due to terrain factors. Tables 5 and 6 present the examined algorithms' results when applied based on vessel AIS data from the MarineCadastre database and the South Channel of the Yangtze River, respectively.

A comparison of the experimental results presented in Tables 4–6 reveals that the DP algorithm is better than IOPW in terms of compression performance and the information integrity of the compressed trajectory. The three tables show that the compression effect of IOPW is close to that of the DP algorithm. In Table 5, when the compression rates of the IOPW and DP algorithms are 23·96% and 6·8%, respectively, the DTW matching distance of the two algorithms is very close. Table 6 shows the same results.

***Table 5.*** *Comparative studies using the MarineCadastre database.*

| | Algorithms | Thresholds | Number | TCR | DTW | RLL | Time (s) |
|---|---|---|---|---|---|---|---|
| Batch | DP | 0·001 | 7,926 | 12·66% | $9 \cdot 292 \times 10^{-3}$ | 0·714% | 2·475 |
| | | 0·003 | 4,208 | 6·72% | $1 \cdot 766 \times 10^{-2}$ | 1·983% | 1·581 |
| | | 0·005 | 3,160 | 5·05% | $2 \cdot 209 \times 10^{-2}$ | 3·098% | 1·334 |
| | | 0·040 | 950 | 1·52% | $4 \cdot 526 \times 10^{-2}$ | 19·107% | 0·589 |
| | US | 3 | 21,130 | 33·76% | $4 \cdot 407 \times 10^{-3}$ | 1·584% | 0·115 |
| | | 7 | 9,285 | 14·84% | $6 \cdot 524 \times 10^{-3}$ | 4·162% | 0·078 |
| | | 10 | 6,614 | 10·57% | $7 \cdot 921 \times 10^{-3}$ | 5·738% | 0·067 |
| | | 60 | 1,443 | 2·31% | $2 \cdot 265 \times 10^{-2}$ | 20·845% | 0·041 |
| Online | SQUISH | 0·01 | 766 | 1·22% | $5 \cdot 135 \times 10^{-2}$ | 37·034% | 2·643 |
| | | 0·04 | 2,495 | 3·99% | $1 \cdot 700 \times 10^{-2}$ | 13·468% | 8·846 |
| | | 0·07 | 4,338 | 6·93% | $1 \cdot 051 \times 10^{-2}$ | 7·095% | 14·138 |
| | | 0·20 | 12,465 | 19·92% | $3 \cdot 865 \times 10^{-3}$ | 1·350% | 32·995 |
| | DR | 0·001 | 12,649 | 20·21% | $7 \cdot 002 \times 10^{-3}$ | 1·087% | 1·174 |
| | | 0·003 | 6,871 | 10·98% | $1 \cdot 066 \times 10^{-2}$ | 3·175% | 1·057 |
| | | 0·005 | 5,101 | 8·15% | $1 \cdot 374 \times 10^{-2}$ | 4·642% | 1·143 |
| | | 0·040 | 1,594 | 2·55% | $3 \cdot 280 \times 10^{-2}$ | 18·304% | 1·110 |
| | OPW | 0·001 | 16,943 | 27·07% | $4 \cdot 270 \times 10^{-3}$ | 0·248% | 3·731 |
| | | 0·003 | 9,275 | 14·82% | $9 \cdot 227 \times 10^{-3}$ | 1·034% | 6·395 |
| | | 0·005 | 6,832 | 10·92% | $1 \cdot 298 \times 10^{-2}$ | 1·585% | 8·109 |
| | | 0·040 | 1,859 | 2·97% | $3 \cdot 085 \times 10^{-2}$ | 8·845% | 17·654 |
| | IOPW | 0·001 | 14,993 | 23·96% | $5 \cdot 141 \times 10^{-3}$ | 0·345% | 7·441 |
| | | 0·003 | 7,646 | 12·22% | $1 \cdot 082 \times 10^{-2}$ | 1·295% | 12·192 |
| | | 0·005 | 5,576 | 8·91% | $1 \cdot 437 \times 10^{-2}$ | 1·976% | 16·138 |
| | | 0·040 | 1,449 | 2·32% | $3 \cdot 535 \times 10^{-2}$ | 10·134% | 35·120 |

The experimental results show that a smaller TCR and smaller RLL result in a better compression effect of the algorithm. Another observation depicted in Figure 8 is that the compression effect of the IOPW algorithm is stronger than the OPW algorithm when the ratio of the linear trajectory and the curved trajectory is standard.

### *5.3. Results evaluation and discussion*

To observe the compression better, AIS vessel trajectory data before and after compression were visualised. As there are differences in the trajectory of each vessel, the compression algorithm must offer durable compression performance. Second, the vessel's course may change within a short period. Generally, this is caused by the need for collision avoidance. This phenomenon is widespread in marine traffic. However, when the vessel changes heading at a considerable angle, compression technology does not have a significant effect. Therefore, the six algorithms of DP, uniform sampling, SQUISH, DR, OPW and IOPW were tested based on the three AIS data sources, as shown in Figures 9–11. The test results were analysed visually. Note that the algorithm parameters were adjusted to obtain a more intuitive contrast. In this analysis, $P_D$, $P_R$, $P_O$ and $P_w$ indicate the thresholds set by the respective algorithm, $P_S$ is the step size, $P_C$ is the reciprocal of the compression rate, and CR indicates the trajectory compression rate. Figures 8(a), 8(b) and 8(c) correspond to Tables 4–6. Since the DP algorithm is an offline algorithm, global optimisation can be achieved and the DP method performs the best for all compression ratios. In Figure 8(c), this difference between US compression results and overall ranking is due to the line in the compression trajectory occupying most of the line that skews the overall average.

*Table 6. Comparative studies for the south channel of the Yangtze river estuary.*

| Arithmetic types | Algorithms | Thresholds | Number of timestamped points | TCR | DTW | RLL | Time (s) |
|---|---|---|---|---|---|---|---|
| Batch | DP | 0·001 | 23,191 | 2·30% | $5\cdot041\times10^{-3}$ | 1·097% | 4·844 |
| | | 0·003 | 12,011 | 1·19% | $9\cdot213\times10^{-3}$ | 2·914% | 3·283 |
| | | 0·005 | 9,595 | 0·95% | $1\cdot078\times10^{-2}$ | 3·701% | 3·000 |
| | | 0·040 | 3,282 | 0·33% | $3\cdot504\times10^{-2}$ | 15·254% | 1·259 |
| | US | 3 | 337,282 | 33·45% | $3\cdot396\times10^{-3}$ | 0·852% | 0·320 |
| | | 7 | 145,582 | 14·44% | $4\cdot069\times10^{-3}$ | 1·796% | 0·188 |
| | | 10 | 102,451 | 10·16% | $4\cdot514\times10^{-3}$ | 2·450% | 0·137 |
| | | 60 | 18,597 | 1·84% | $1\cdot016\times10^{-2}$ | 8·280% | 0·079 |
| Online | SQUISH | 0·01 | 10,148 | 1·01% | $1\cdot824\times10^{-2}$ | 15·400% | 32·551 |
| | | 0·04 | 40,060 | 3·97% | $6\cdot356\times10^{-3}$ | 6·467% | 109·370 |
| | | 0·07 | 70,219 | 6·96% | $3\cdot544\times10^{-3}$ | 4·019% | 194·644 |
| | | 0·20 | 201,310 | 19·97% | $9\cdot716\times10^{-4}$ | 1·358% | 449·269 |
| | DR | 0·001 | 41,037 | 4·07% | $5\cdot570\times10^{-3}$ | 1·371% | 4·232 |
| | | 0·003 | 21,512 | 2·13% | $8\cdot129\times10^{-3}$ | 3·223% | 4·205 |
| | | 0·005 | 15,850 | 1·57% | $9\cdot965\times10^{-3}$ | 4·699% | 3·203 |
| | | 0·040 | 5,442 | 0·54% | $2\cdot197\times10^{-2}$ | 14·355% | 3·242 |
| | OPW | 0·001 | 39,054 | 3·87% | $3\cdot966\times10^{-3}$ | 0·773% | 55·10 |
| | | 0·003 | 20,032 | 1·99% | $7\cdot349\times10^{-3}$ | 1·770% | 90·522 |
| | | 0·005 | 14,269 | 1·42% | $9\cdot184\times10^{-3}$ | 2·723% | 113·435 |
| | | 0·040 | 4,151 | 0·41% | $3\cdot147\times10^{-2}$ | 10·730% | 587·956 |
| | IOPW | 0·001 | 30,117 | 2·99% | $5\cdot106\times10^{-3}$ | 1·044% | 137·010 |
| | | 0·003 | 15,640 | 1·55% | $8\cdot642\times10^{-3}$ | 2·483% | 213·982 |
| | | 0·005 | 11,781 | 1·17% | $1\cdot061\times10^{-2}$ | 3·477% | 261·348 |
| | | 0·040 | 3,379 | 0·34% | $3\cdot792\times10^{-2}$ | 13·314% | 691·664 |

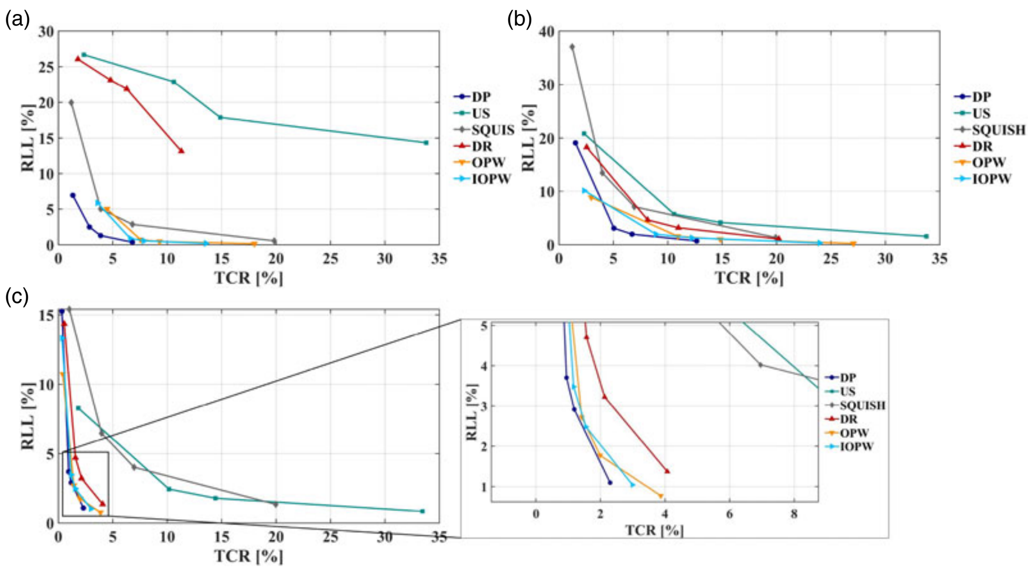


*Figure 8. Values of RLL and TCR under different PED thresholds.*

https://doi.org/10.1017/S0373463324000171 Published online by Cambridge University Press


***Figure 9.*** *Experiment results based on AIS data of Wuhan Section of Yangtze River (August 2016).*

Figure 9 shows that some AIS information points with large-angle steering and certain representativeness cannot be compressed in the algorithm compression process. When $\mathrm{P_C} = 0 \cdot 01$, the trajectory structure is destroyed. As $\mathrm{P_C}$ gradually increases, the trajectory structure begins to form after compression by the SQUISH algorithm. When $\mathrm{P_C} = 0 \cdot 07$, the compressed trajectory structure is higher than the original trajectory. Similarly, the uniform algorithm does not maintain the trajectory integrity very well. When $\mathrm{P_S} = 10$, the trajectory structure changes significantly compared with the original trajectory, especially the second half of the trajectory. When the $\mathrm{P_D}$, $\mathrm{P_R}$ and $\mathrm{P_O}$ thresholds of the other three algorithms are between $0 \cdot 001$ and $0 \cdot 005$, the compressed trajectory structure undergoes significant deformation compared with the original trajectory.

Figure 10 indicates that when the $P_D$, $P_R$, $P_W$ and $P_O$ of the DP, DR, OPW and IOPW algorithms are set to the maximum value of $0 \cdot 04$, the trajectory structures given by the DP and IOPW algorithms exhibit less similarity to the original trajectory. The trajectory structure provided by the DR algorithm indicates a more significant change than the initial trajectory, but it is superior to the DP and IOPW algorithms. Therefore, the DR algorithm is more adaptable to the heading change trajectory than the DP and IOPW algorithms. For the US algorithm, the points saved after compression are more regular, and this algorithm takes points at equal intervals. Thus, there are a few cases in which some representative AIS information points are discarded. When $P_S = 3$, $\mathrm{CR} = 33 \cdot 76\%$, the algorithm retains more points after compression. However, the compressed trajectory structure begins to deform.

Figure 11 indicates that when $P_D$ and $P_O$ are set to $0 \cdot 04$, the compression rate of the DP algorithm is close to that of IOPW and the compressed trajectory of the two algorithms maintain a good degree of completeness compared with the original image. However, the time complexity of the DP algorithm is higher than those of the other four algorithms, as the DP algorithm implements a process in which decomposition methods are recursively called. This provides the DP algorithm with a high degree of time complexity. When $P_C = 0 \cdot 01$ and $P_S = 60$, the compression rates of the SQUISH and US algorithms are higher, but the integrity of the trajectory after compression is lower and the information loss is more serious.

Finally, in Figures 9–11, the compressed trajectory of the IOPW algorithm has higher similarity than the original trajectory. The IOPW algorithm can be selected in trajectory data compression when parallel computation is involved. In addition, the IOPW algorithm is the best choice when high compression with good data fidelity is needed in online applications, where low error tolerance and some latencies are acceptable.

In summary, the data compression error of the IOPW algorithm is better than other online algorithms in the experiments. It is imperative to acknowledge that its running time is comparatively higher than

https://doi.org/10.1017/S0373463324000171 Published online by Cambridge University Press

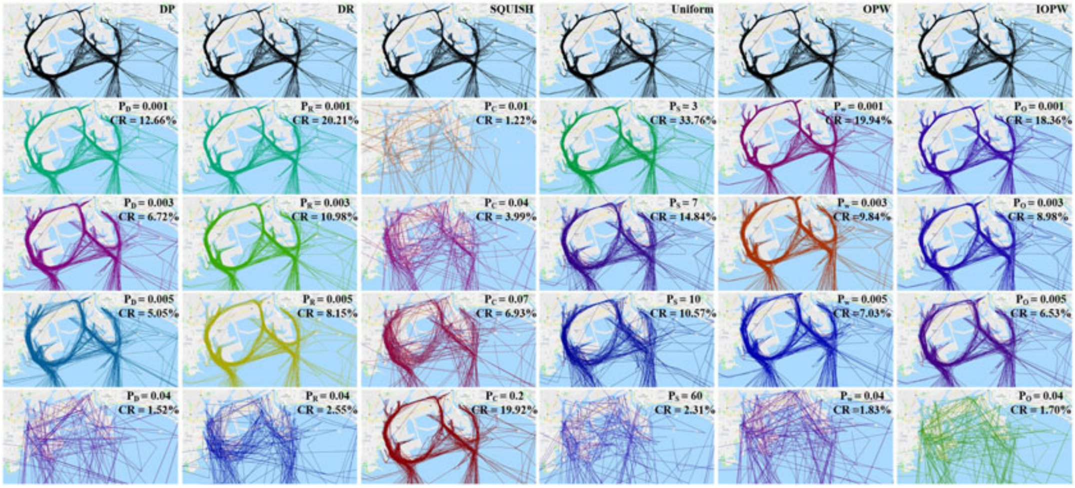


***Figure 10.*** *Experimental results based on AIS data of MarineCadastre database (April 2016).*

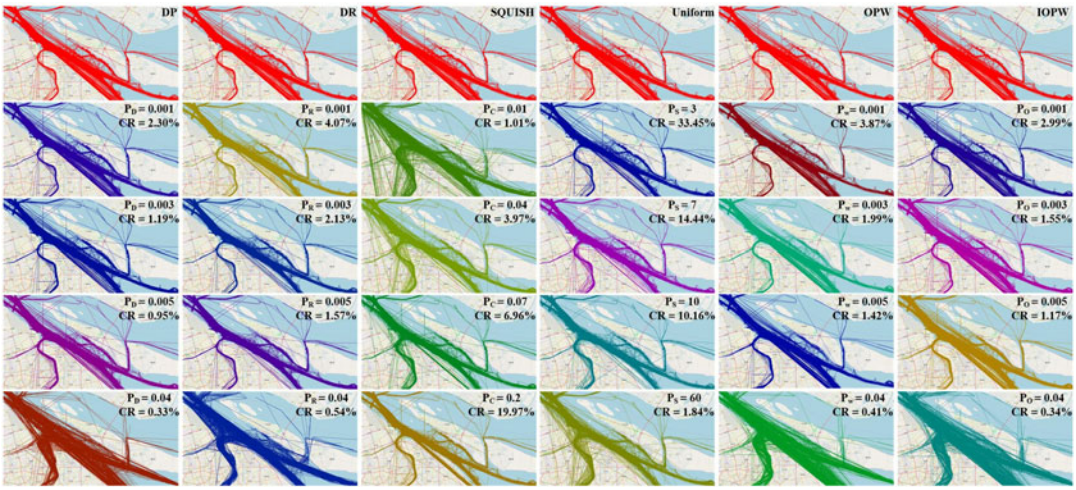


***Figure 11.*** *Experimental results based on AIS data of South Channel of Yangtze River (August 2017).*

that of other online compression algorithms. This discrepancy in execution times can be attributed to the elevated time complexity inherent in the IOPW algorithm. Despite its prolonged running time, the algorithm's superior compression performance underscores the trade-off between time complexity and compression effectiveness. The data compression error caused by the DP algorithm is minimal, but it does not support real-time processing. Although the error caused by the IOPW algorithm is a little larger than that by the DP algorithm, it supports real-time processing. Therefore, when one needs to compress historical ship trajectory data, the Douglas–Peucker algorithm is recommended, and when one needs to compress ship trajectory data in real-time, the IOPW algorithm is recommended.

## 6. Conclusions and future work

In this paper, an online vessel trajectory compression algorithm based on IOPW is introduced. This algorithm could effectively compress the massive vessel trajectories and reduce the rate of distance loss. The significant contributions of this paper are twofold. First, a weighted Euclidean distance by fusing traditional PED and SED was incorporated into the conventional OPW algorithm. The main benefit of

https://doi.org/10.1017/S0373463324000171 Published online by Cambridge University Press

this weighted distance is that it takes full advantage of both spatial and temporal properties of AIS-based vessel trajectories. Thus, the rate of distance loss could be reduced while preserving the essential trajectory features. Second, the proposed IOPW algorithm can implement online compression of vessel trajectories, which is significantly different from the offline compression methods.

The results of the real trajectory experiments showed that the IOPW algorithm was approximately equal to or slightly smaller than the classical DP algorithm in terms of the compression rate. The performance indices of DTW, RLL, PED and SED were optimised. In other words, the IOPW algorithm compressed the trajectory more closely to the original route while maintaining the high compression rate of the classical DP algorithm. Multi-turn or circling trajectories often exist in inland or harbour water areas. Therefore, in the research of maritime traffic and related fields in these waters, the IOPW algorithm can be used as a preprocessing method for the AIS data, which can retain more critical information than what must be extracted from the route. In conclusion, when applied to vessel AIS trajectory data compression, it enhances the operational efficiency of computers. Furthermore, it establishes a theoretical foundation for future processing of vessel AIS trajectory big data, ship motion pattern recognition, ship motion feature extraction and research on the behaviour of single vessels.

The favourable compression outcomes achieved are widely attributed to the implementation of the IOPW algorithm with weighted distance. Although this proposed method enhances compression performance, it does come at a higher computational cost. Using modern graphics processing units (GPUs) as prevalent and cost-effective parallel computing platforms is well recognised in reducing the computational time required for trajectory data mining. Therefore, there is a great potential to further accelerate our IOPW for real-time online compression in the GPU computing platform in future work.

**Acknowledgment.** This research of Z. Liu is supported though the National Natural Science Foundation of China (No.: 52171351).

## References

**Cao, H., Wolfson, O. and Trajcevski, G.** (2006). Spatio-temporal data reduction with deterministic error bounds. *VLDB Journal*, **15**(3), 211–228.

**Cao, X., Cong, G. and Jensen, C. S.** (2010). Mining significant semantic locations from GPS data. *Proceedings of the VLDB Endowment*, **3**(1-2), 1009–1020.

**Douglas, D. H. and Peucker, T. K.** (1973). Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica: The International Journal for Geographic Information and Geovisualization*, **10**(2), 112–122.

**Ducruet, C.** (Ed.) (2015). *Maritime Networks: Spatial Structures and Time Dynamics*. New York, NY: Routledge.

**Fu, Z., Hu, W. and Tan, T.** (2005). Similarity Based Vehicle Trajectory Clustering and Anomaly Detection. In *IEEE International Conference on Image Processing 2005*, Vol. **2**, pp. II-602. IEEE.

**García, J., Concha, O. P., Molina, J. M. and De Miguel, G.** (2006). Trajectory Classification Based on Machine-Learning Techniques Over Tracking Data. In *2006 9th International Conference on Information Fusion*, pp. 1–8. IEEE.

**Huang, Y., Li, Y., Zhang, Z. and Liu, R.** (2020). GPU-accelerated compression and visualization of large-scale vessel trajectories in maritime IoT industries. *IEEE Internet of Things Journal*, **7**(11), 10794–10812.

**Kellaris, G., Pelekis, N. and Theodoridis, Y.** (2013). Map-matched trajectory compression. *Journal of Systems and Software*, **86**(6), 1566–1579.

**Keogh, E. and Ratanamahatana, C. A.** (2005). Exact indexing of dynamic time warping. *Knowledge and Information Systems*, **7**(3), 358–386.

**Keogh, E., Chu, S., Hart, D. and Pazzani, M.** (2001). An Online Algorithm for Segmenting Time Series. In *Proceedings 2001 IEEE International Conference on Data Mining*, pp. 289–296. IEEE.

**Le Tixerant, M., Le Guyader, D., Gourmelon, F. and Queffelec, B.** (2018). How can automatic identification system (AIS) data be used for maritime spatial planning? *Ocean Coastal Manage*, **166**, 18–30.

**Li, H. and Yang, Z.** (2023). Incorporation of AIS data-based machine learning into unsupervised route planning for maritime autonomous surface ships. *Transportation Research Part E: Logistics and Transportation Review*, **176**, 103171.

**Li, H., Liu, J. and Liu, R. W.** (2017). Dynamic Speed Control Model for Very Large Ship in Restricted Waters. In *The 27th International Ocean and Polar Engineering Conference*.

**Li, H., Lam, J. S. L., Yang, Z., Liu, J., Liu, R. W., Liang, M. and Li, Y.** (2022). Unsupervised hierarchical methodology of maritime traffic pattern extraction for knowledge discovery. *Transportation Research. Part C: Emerging Technologies*, **143**, 103856.

**Liu, X., Cheung, Y. M., Peng, S. J., Cui, Z., Zhong, B. and Du, J. X.** (2014). Automatic motion capture data denoising via filtered subspace clustering and low rank matrix approximation. *Signal Process*, **105**, 350–362.

**Liu, J., Zhao, K., Sommer, P., Shang, S., Kusy, B. and Jurdak, R.** (2015). Bounded Quadrant System: Error-Bounded Trajectory Compression on the go. In *2015 IEEE 31st International Conference on Data Engineering*, pp. 987–998. IEEE.

**Liu, J., Zhou, F., Li, Z., Wang, M. and Liu, R. W.** (2016). Dynamic ship domain models for capacity analysis of restricted water channels. *Journal of Navigation*, **69**(3), 481–503.

**Liu, J., Li, H., Yang, Z., Wu, K., Liu, Y. and Liu, R. W.** (2019). Adaptive Douglas-Peucker algorithm with automatic thresholding for AIS-based vessel trajectory compression. *IEEE Access*, **7**, 150677–150692.

**Liu, J., Shi, G. and Zhu, K.** (2020). Online multiple outputs least-squares support vector regression model of ship trajectory prediction based on automatic information system data and selection mechanism. *IEEE Access*, **8**, 154727–154745.

**Makris, A., Tserpes, K. and Anagnostopoulos, D.** (2019a). Database System Comparison Based on Spatiotemporal Functionality. *Proceedings of the 23rd International Database Applications & Engineering Symposium*, pp. 1–7.

**Makris, A., Tserpes, K. and Spiliopoulos, G.** (2019b). Performance Evaluation of MongoDB and PostgreSQL for Spatio-temporal Data[C]//EDBT/ICDT Workshops.

**Makris, A., Tserpes, K. and Spiliopoulos, G.** (2021). MongoDB Vs PostgreSQL: A comparative study on performance aspects. *GeoInformatica*, **25**(2), 243–268.

**Meratnia, N. and Rolf, A.** (2004). Spatiotemporal Compression Techniques for Moving Point Objects. In *International Conference on Extending Database Technology*, pp. 765–782. Berlin, Heidelberg: Springer.

**Muckell, J., Hwang, J. H., Lawson, C. T. and Ravi, S. S.** (2010). Algorithms for Compressing GPS Trajectory Data: An Empirical Evaluation. In *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pp. 402–405.

**Muckell, J., Hwang, J. H., Patil, V., Lawson, C. T., Ping, F. and Ravi, S. S.** (2011). SQUISH: An Online Approach for GPS Trajectory Compression. In *Proceedings of the 2nd International Conference on Computing for Geospatial Research & Applications*, pp. 1–8.

**Muckell, J., Olsen, P. W., Hwang, J. H., Lawson, C. T. and Ravi, S. S.** (2014). Compression of trajectory data: A comprehensive evaluation and new approach. *GeoInformatica*, **18**(3), 435–460.

**Nguyen, D., Vadaine, R., Hajduch, G., Garello, R. and Fablet, R.** (2018). A Multi-Task Deep Learning Architecture for Maritime Surveillance Using ais Data Streams. In *2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 331–340.

**Qi, L. and Ji, Y.** (2020). Ship trajectory data compression algorithms for automatic identification system: Comparison and analysis. *Journal of Water Resources and Ocean Science*, **9**(2), 42–47.

**Sang, L. Z., Wall, A., Mao, Z., Yan, X. P. and Wang, J.** (2015). A novel method for restoring the trajectory of the inland waterway ship by using AIS data. *Ocean Eng*, **110**, 183–194.

**Shi, W. and Cheung, C.** (2006). Performance evaluation of line simplification algorithms for vector generalization. *The Cartographic Journal*, **43**(1), 27–44.

**Trajcevski, G., Cao, H., Scheuermanny, P., Wolfsonz, O. and Vaccaro, D.** (2006). On-line Data Reduction and the Quality of History in Moving Objects Databases. In *Proceedings of the 5th ACM International Workshop on Data Engineering for Wireless and mobile Access*, pp. 19–26.

**van Emde Boas, P.** (1975). Preserving Order in A Forest in Less Than Logarithmic Time. In *16th Annual Symposium on Foundations of Computer Science*, pp. 75–84. IEEE.

**Wang, X., Tieu, K. and Grimson, E.** (2006). Learning Semantic Scene Models by Trajectory Analysis. In *European Conference on Computer Vision*, pp. 110–123. Berlin, Heidelberg: Springer.

**Wang, D., Miwa, T. and Morikawa, T.** (2020). Big trajectory data mining: A survey of methods, applications, and services. *Sensors*, **20**(16), 4571.

**Wang, X., Liu, Z., Yan, R., Wang, H. and Zhang, M.** (2022). Quantitative analysis of the impact of COVID-19 on ship visiting behaviors to ports-A framework and a case study. *Ocean & Coastal Management*, **230**, 106377.

**Wei, Z., Zhou, K., Wei, M. and Shi, G.** (2016). Ship motion pattern recognition and application based on AIS data. *Journal of Shanghai Maritime University*, **37**(2), 17–22. (in Chinese).

**Weng, J., Shi, K., Gan, X., Li, G. and Huang, Z.** (2020). Ship emission estimation with high spatial-temporal resolution in the Yangtze river estuary using AIS data. *Journal of Cleaner Production*, **248**, 119297.

**Wu, X., Mehta, A. L., Zaloom, V. A. and Craig, B. N.** (2016). Analysis of waterway transportation in southeast Texas waterway based on AIS data. *Ocean Eng*, **121**, 196–209.

**Xiao, Z., Ponnambalam, L., Fu, X. and Zhang, W.** (2017). Maritime traffic probabilistic forecasting based on vessels' waterway patterns and motion behaviors. *IEEE Transactions on Intelligent Transportation Systems*, **18**(11), 3122–3134.

**Zhang, S. K., Shi, G. Y., Liu, Z. J., Zhao, Z. W. and Wu, Z. L.** (2018). Data-driven based automatic maritime routing from massive AIS trajectories in the face of disparity. *Ocean Engineering*, **155**, 240–250.

**Zhang, M., Montewka, J., Manderbacka, T., Kujala, P. and Hirdaris, S.** (2021). A big data analytics method for the evaluation of ship-ship collision risk reflecting hydrometeorological conditions. *Reliability Engineering & System Safety*, **213**, 107674.

**Zhang, M., Kujala, P. and Hirdaris, S.** (2022). A machine learning method for the evaluation of ship grounding risk in real operational conditions. *Reliability Engineering & System Safety*, **226**, 108697.

**Zhang, M., Kujala, P., Musharraf, M., Zhang, J. and Hirdaris, S.** (2023). A machine learning method for the prediction of ship motion trajectories in real operational conditions. *Ocean Engineering*, **283**, 114905.

https://doi.org/10.1017/S0373463324000171 Published online by Cambridge University Press

**Zhang, M., Tsoulakos, N., Kujala, P. and Hirdaris, S.** (2024). A deep learning method for the prediction of ship fuel consumption in real operational conditions. *Engineering Applications of Artificial Intelligence*, **130**, 107425.
**Zhao, L. and Shi, G.** (2018). A method for simplifying ship trajectory based on improved Douglas–Peucker algorithm. *Ocean Eng*, **166**, 37–46.
**Zhao, L. and Shi, G.** (2019). A trajectory clustering method based on Douglas–Peucker compression and density for marine traffic pattern recognition. *Ocean Eng*, **172**, 456–467.
**Zhao, L., Ochieng, W. Y., Quddus, M. A. and Noland, R. B.** (2003). An extended Kalman filter algorithm for integrating GPS and low cost dead reckoning system data for vehicle performance and emissions monitoring. *Journal of Navigation*, **56**(2), 257–275.
**Zhao, L., Shi, G. and Yang, J.** (2018a). Ship trajectories pre-processing based on AIS data. *Journal of Navigation*, **71**(5), 1210–1230.
**Zhao, Y., Shang, S., Wang, Y., Zheng, B., Nguyen, Q. V. H. and Zheng, K.** (2018b). REST: A Reference-Based Framework for Spatio-Temporal Trajectory Compression. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 2797–2806.
**Zheng, Y.** (2015). Trajectory data mining: An overview. *ACM Transactions on Intelligent Systems and Technology*, **6**(3), 1–41.
**Zhou, Z., Zhang, Y., Yuan, X. and Wang, H.** (2023). Compressing AIS trajectory data based on the multi-objective peak Douglas–Peucker algorithm. *IEEE Access*, **11**, 6802–6821.

**Cite this article:** Liu Z, Yuan W, Liang M, Zhang M, Liu C, Liu RW, Liu J (2024). An online method for ship trajectory compression using AIS data. *The Journal of Navigation* 1–22. https://doi.org/10.1017/S0373463324000171