Guo, Shaoqing; Bolbot, Victor; Valdez Banda, Osiris

An adaptive trajectory compression and feature preservation method for maritime traffic analysis

Published in:
Ocean Engineering

DOI:
10.1016/j.oceaneng.2024.119189

Published: 15/11/2024

Document Version
Publisher's PDF, also known as Version of record

Ocean Engineering 312 (2024) 119189

Contents lists available at ScienceDirect

Ocean Engineering

journal homepage: www.elsevier.com/locate/oceaneng

Research paper

An adaptive trajectory compression and feature preservation method for maritime traffic analysis

Shaoqing Guo [a,b,*], Victor Bolbot [a,b], Osiris Valdez Banda [a,b]

[a] Department of Mechanical Engineering, Marine Technology, Research Group on Safe and Efficient Marine and Ship Systems, Aalto University, Espoo, Finland
[b] Kotka Maritime Research Centre, Kotka, Finland

A R T I C L E I N F O

Keywords:
Adaptive ship trajectory compression
Feature preservation
AIS data
Top-down kinematic compression
Data-driven analysis
Douglas-peucker algorithm

A B S T R A C T

Ship trajectory data extracted from Automatic Identification System (AIS) has been extensively used for maritime traffic analysis. Yet the enormous volume of AIS data has come with substantial challenges related to storing, processing, analyzing, transmitting, and transferring. Trajectory compression techniques have been widely investigated to remedy the challenge. However, conventional compression techniques such as Douglas-Peucker (DP) algorithm mainly depend on line simplification algorithms, falling short in accurately identifying and preserving crucial information within trajectories. Moreover, using kinematic information from AIS data has posed difficulties associated with compression threshold determination. Hence, an adaptive method capable of considering multiple information from AIS is required. In this paper, a Top-Down Kinematic Compression (TDKC) algorithm aimed at adaptive trajectory compression and feature preservation is proposed. By incorporating time, position, speed, and course attributes from AIS data, TDKC exploits a Compression Binary Tree (CBT) method to address the recursion termination problem and determine the threshold automatically. A case study was conducted to evaluate the performance of TDKC using AIS data from Gulf of Finland, where a comparison with conventional algorithms and their improved versions based on specific performance evaluation metrics was involved. The results demonstrate TDKC's superiority in facilitating maritime traffic analysis.

1. Introduction

Maritime industry is committed to improving safety and security alongside improvements in financial efficiency and greater environmental protection (Tavakoli et al., 2023; EMSA, 2019). Since 2002, Automatic Identification System (AIS) transponders have become mandatory on all passenger and cargo ships with Gross Tonnage over 300 tons, according to the revision of SOLAS (IMO, 1974). The installation of AIS transponders enables ships to transmit own ship information and receive similar data from nearby ships every 2–180 s, as well as the same data transmission to the shore-based stations monitoring traffic conditions (Yang et al., 2019). The communicated information consists of dynamic and static types. Dynamic information includes position, speed over ground (SOG), course over ground (COG), heading, etc., whilst static information includes Maritime Mobile Service Identity (MMSI), ship type, size, draught, etc (ITU, 2014). Although AIS was initially designed to strengthen navigational safety (Svanberg et al., 2019), the majority of maritime traffic models have primarily utilized AIS trajectory data (Zhou et al., 2019). The advancements in big data analytics and artificial intelligence have empowered researchers with the ability to explore more extensively the maritime traffic in connection to accident prevention (Chen et al., 2019; Guo et al., 2023), environmental impact (Bencs et al., 2020), shipping monitoring and management (Li and Ren, 2022; Yin et al., 2022), automation and remote control (Bolbot et al., 2022). Simultaneously, though, the exponential growth in available AIS data has brought challenges in data storing, processing, analyzing, transmitting, and transferring (Sun et al., 2020). Considering that ship motion is often stable, redundant information can be removed from trajectory data, which opens opportunities for AIS trajectory data compression (Zhang et al., 2018).

Trajectory compression has been initially explored within the Geographic Information System (GIS) field in relation to road traffic before being extensively applied in maritime field (Sandu Popa et al., 2015). Relevant techniques can be classified as either lossless or lossy (Han et al., 2017). A lossless compression algorithm aims to reduce the data size without any loss of information through meticulous

* Corresponding author. Department of Mechanical Engineering, Marine Technology, Research Group on Safe and Efficient Marine and Ship Systems, Aalto University, Espoo, Finland.
E-mail addresses: shaoqing.guo@aalto.fi, gsqnbsr@gmail.com (S. Guo).

https://doi.org/10.1016/j.oceaneng.2024.119189
Received 13 June 2024; Received in revised form 26 August 2024; Accepted 3 September 2024
Available online 9 September 2024

optimization of data formats and indexing techniques (Ziv and Lempel, 1978). In contrast, a lossy compression algorithm eliminates redundant, non-essential information to achieve higher desirable compression ratios at an acceptable quality (Singh et al., 2017). In maritime research, where the focal point of investigation always revolves around the ship itself, lossy compression algorithms have found extensive applications due to their flexible adjustment to achieve higher compression rates (Du et al., 2021a; Zhang et al., 2022). Therefore, the literature review of this study mainly concentrates on lossy compression methods, which will be elaborated in Section 2.

Conventional trajectory compression methods predominantly emphasize condensing the information presented by positional attributes through line simplification algorithms (Bellman, 1961). They have benefited a spectrum of applications related to data-driven analysis, such as ship behavior analysis (Li, 2021), path planning (Du et al., 2021b) and traffic pattern recognition (Zhao and Shi, 2019). However, the ignorance of other kinematic attributes including ship speed and course has posed challenges in accurately preserving essential ship trajectory characteristics (Liu et al., 2023). Hence, it is essential to develop a compression method capable of retaining key points by comprehensively considering kinematic information in AIS data. This paper proposes a novel trajectory compression method by leveraging multiple kinematic attributes from AIS data. The proposed method is analytically verified to effectively compress trajectory data while preserving key features, outperforming previous methods applied in maritime traffic analysis.

The remainder of this article is organized as follows. Section 2 presents a comprehensive literature review of trajectory compression methods and their applications in maritime. Section 3 illustrates the key characteristics of the proposed method, highlights its differences from conventional approaches, and defines the metrics for performance evaluation. Section 4 introduces the experimental background of a case study using AIS data from Gulf of Finland. In Section 5, the results are presented and analyzed, wherein the proposed algorithm is compared against the other widely employed compression algorithms and their improved versions. The derived results are critically discussed with the support of a scoring system. Finally, Section 6 summarizes the main research findings and limitations.

2. Literature review

To identify redundant information within trajectory points, trajectory compression algorithms commonly comprise two steps: traversal and simplification (Amigo et al., 2021). The traversal step determines how points in the trajectory are examined (Sun et al., 2016) and the information difference between points is evaluated using specific measurement methods in the simplification step (Ke et al., 2016). Information difference refers to dynamic changes in the spatiotemporal characteristics of the moving object (Zhong et al., 2022). In practice, the high sampling rate of trajectory data usually renders the information difference between consecutive points unapparent (Tang et al., 2019). Similarly, the information in stationary points also remains relatively unchanged (Liu et al., 2021). Therefore, a trajectory point that manifests significant information changes is designated as a key point containing indispensable information that ought to be preserved; otherwise, it can be disregarded (Amigo et al., 2022).

As shown in Fig. 1, compression algorithms are categorized into three types based on traversal search methods: top-down, bottom-up, and sliding window. A top-down algorithm is typically designed to run in an offline mode, which is also known as batch mode, requiring the entire trajectory as input for execution (Meratnia and de By, 2003). It identifies one or several key points necessary to split the trajectory, and for each segment repeats the process until no further key points are found. Well-known top-down algorithms include Douglas-Peucker (DP) algorithm (Douglas and Peucker, 1973) and its improved version Top-Down Time-Ratio (TD-TR) algorithm (Meratnia and de By, 2004).

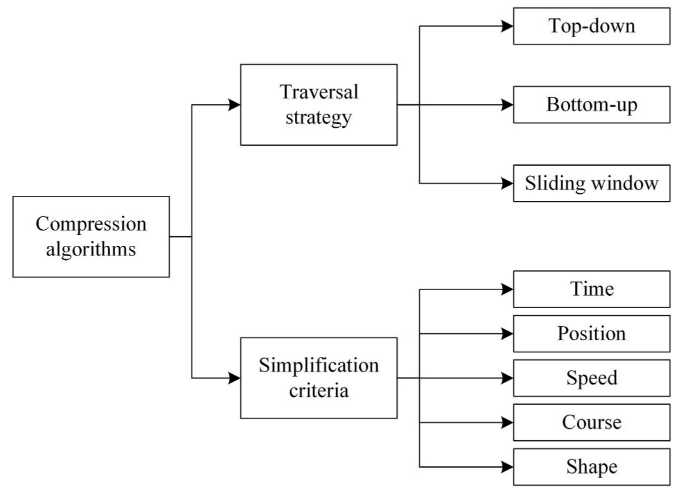


Fig. 1. Categories of compression algorithms.

However, the recursive nature renders top-down algorithms time-consuming. Hence, the optimization of top-down algorithms in terms of traversal has primarily concentrated on reducing the number of recursions required (Zhao and Shi, 2018).

Conversely, bottom-up and sliding window algorithms are recognized as operating online since they analyze points sequentially and handle incoming data progressively (Makris et al., 2021b). A bottom-up algorithm considers an initial segment as a reference, continuously processes subsequent points to merge the segments and updates the reference segment until all points are examined. The representative algorithms include Spatiotemporal Trace (STTrace) algorithm, Spatial Quality Simplification Heuristic (SQUISH) algorithm and Dead Reckoning algorithm (Potamias et al., 2006; Muckell et al., 2011, 2014; Trajcevski et al., 2006). Since conventional bottom-up algorithms utilize local features of the entire trajectory, the enhancement of their global view was implemented by graph-based traverse strategies proposed in Multiresolution Polygonal Approximation (MRPA) and Directed acyclic graph based Online Trajectory Simplification (DOTS) algorithms (Chen et al., 2012; Cao and Li, 2017). Moreover, in a sliding window algorithm, a window with a certain size is maintained to seamlessly traverse the entire trajectory. For each sliding, only points in the window are handled to identify key points. A classical sliding window algorithm can be found in (Keogh et al., 2004). Instead of setting a constant value, dynamic window size configuration has demonstrated an evolutionary improvement flexible to the trajectory motion characteristics. This technique is often referred to as Opening Window (Meratnia and de By, 2004).

Conventional algorithms primarily measure information differences based on positional data during the simplification step (Liu et al., 2024). Their intuitiveness and ease of implementation render themselves applicable to various maritime research studies (Lee et al., 2022; Rong et al., 2020). With advancements in modern sensing and processing technologies enabling the capture of motion data such as speed and course, improvements are allowed in information difference measurement compared to traditional methods (Qian and Lu, 2017). Examples include the measurement of speed and course changes in online algorithms (Zhu and Ma, 2021; Zhang et al., 2020). Meanwhile, since top-down algorithms gradually measure information differences from larger to smaller temporal scales, emphasis is not only on changes in speed or course (De Vries and Van Someren, 2012), but also on discerning shape variations within trajectories. For instance, transitions in distinct segments were identified in (Wu and Pelot, 2007). Besides, in some studies, attributes were also aggregated to provide more accurate information difference measurement instead of independent assessment (Shi and Liu, 2022). The perspectives of criteria for simplification are shown in Fig. 1.

Traditional algorithms inherently encounter difficulties in setting thresholds for key point identification, which are further compounded by the incorporation of multi-dimensional measurements in advancing algorithms (Sanchez-Heres, 2019). For example, position, speed, and course thresholds were manually configured with a diversity of values in different studies (Wang, 2013). In addition, aggregated measurement necessitates proper normalization of the attributes and determination of their respective weights (Zhou et al., 2023b). Investigation has also revealed that using identical thresholds across different trajectories will discard or maintain features inappropriately (Liu et al., 2019). Within this context, adaptive threshold setting has emerged as a promising alternative as demonstrated in (Ji et al., 2022), where a dynamic grating algorithm was developed for AIS data compression purposes.

Given the capability of top-down algorithms to consider trajectory information from a global perspective, numerous adaptive threshold setting strategies have been proposed within the context of these algorithms (Zhang et al., 2016). However, they lack robustness and face various challenges. To illustrate, statistical methods have been proposed to support the threshold determination (Huang et al., 2020), yet, they are complicated and require adaptation to the traffic area change. The selection of threshold candidates for testing in statistical analysis also suffers from discreteness, which fails to guarantee that the chosen thresholds are necessarily optimal (Gao and Shi, 2019). In other approaches, additional data not included in the trajectory such as ship size, engine emissions, obstacles presence, and coastline information has supported the thresholds setting (Gao et al., 2023; Gu et al., 2023; Lee and Cho, 2022; Wei et al., 2020), but these kinds of information are not always available, leading to their limited applicability. Moreover, algorithmic instability can arise from the utilization of different coordinate systems as in (Li et al., 2022) for example, where the slope is used to adaptively calculate the threshold without specifying a certain coordinate system. When different coordinate systems are utilized, the compression results are not identical. Finally, the recursion termination problem that commonly exists in top-down algorithms still remains unresolved, resulting in the premature ending of the compression process due to inappropriate threshold settings (Tang et al., 2021).

Meanwhile, the employment of multiple attributes has rendered performance evaluation lacking in comprehensiveness (Makris et al., 2021a). According to previous studies, the predominant performance metrics include running time (RT), compression rate (CR), and length loss rate (LLR) (Yan et al., 2022). These metrics serve as key indicators for evaluating the efficiency and effectiveness of AIS compression algorithms. In addition, distance-based similarity (SMD) measurement method functions as another important metric for evaluating compression performance (Sousa et al., 2021). However, the absence of metrics related to speed and course hinders the comprehensive assessment of algorithm performance. Although a speed preservation metric was proposed in (Leichsenring and Baldo, 2020), utilizing average speed as a metric overlooks speed variations and distributions, leading to potentially inaccurate evaluations. Therefore, to provide a more holistic evaluation, it is imperative to introduce metrics concerning speed and course.

In this study, we have summarized advanced compression algorithms proposed in maritime field as presented in Table 1. It is evident that top-down algorithm has become the prevailing approach. However, recent studies have rarely considered time, position, speed, and course simultaneously during compression. The one considering them was proposed by Zhou et al. (2023b) without adaptive threshold determination. Meanwhile, advancing trajectory compression research mainly focuses on the comparison with traditional algorithms, while comparative studies among emerging algorithms remain relatively limited (Yan et al., 2022). Therefore, in this paper, we propose a novel kinematic-based approach named Top-Down Kinematic Compression (TDKC) and compare it with various compression techniques that consider different attributes from AIS data. In contrast to the previous top-down

Table 1
Trajectory compression methods in the maritime field.

Literature	Type of approach	Attribute in trajectories considered	Threshold determination	Additional information required for threshold determination
De Vries and Van Someren (2012)	Top-down	Time, position, speed	Manual	/
Du et al. (2021b)	Top-down	Position	/	Obstacle
Gao et al. (2023)	Top-down	Time, position, speed	Manual	Engine, emission
Gu et al. (2023)	Top-down	Position	Adaptive	Obstacle
Lee and Cho (2022)	Top-down	Position	Statistical	Coastline
Li et al. (2022)	Top-down	Time, position, speed	Adaptive	/
Liu et al. (2023)	Top-down	Position, speed, course	Statistical	/
Liu et al. (2019)	Top-down	Position	Adaptive	/
Sanchez-Heres (2019)	Top-down	Time, position	/	/
Shi and Liu (2022)	Top-down	Time, position, speed	Manual, statistical	/
Tang et al. (2021)	Top-down	Position	Adaptive	/
Wu and Pelot (2007)	Top-down	Position	Manual	/
Zhang et al. (2016)	Top-down	Position	Adaptive	Ship length
Zhao and Shi (2019)	Top-down	Position	Statistical	/
Zhao and Shi (2018)	Top-down	Time, position, course	Statistical, manual	Ship length
Zhou et al. (2023b)	Top-down	Time, position, speed, course	Manual	/
Ji et al. (2022)	Bottom-up	Position	Statistical	/
Ma et al. (2022)	Bottom-up	Position	/	/
Zhang et al. (2020)	Bottom-up	Time, position, speed, course	Manual	/
Gao and Shi (2019)	Sliding window	Time, position, course	Statistical	Ship width
Li and Ren (2022)	Sliding window	Position	Manual	/
Liu et al. (2024)	Sliding window	Time, position	Statistical	/
Sun et al. (2020)	Sliding window	Position	Statistical	/
Zhu and Ma (2021)	Sliding window	Time, speed, course	Manual	/
Wei et al. (2020)	Top-down and sliding window	Position, speed, course	Statistical	Ship length
Yan et al. (2022)	Sliding window and bottom-up	Time, position	Statistical	Coastline

approaches, TDKC incorporates the kinematic information in AIS including timestamp, longitude, latitude, SOG, and COG to compress the ship trajectory data. Additional novel contributions include solving the recursion termination problem through a Compression Binary Tree (CBT) and an adaptive threshold setting strategy development, which balances between compression rate and feature preservation, avoiding inappropriate threshold configurations. To accelerate the processing speed of CBT construction, a new running time optimization strategy is further proposed. Along with common metrics, a new metric indicating velocity-based similarity (SMV) using Dynamic Time Warping (DTW) method is proposed. Furthermore, the metrics adopted in this paper are converted into semi-quantitative measures to establish a scoring system that facilitates the comparison of different algorithms.

3. Methods

TDKC algorithm is a typical top-down algorithm which improves from Douglas-Peucker (DP) algorithm as presented in (Douglas and Peucker, 1973). In this section, we first introduce the basic DP algorithm. Then, the procedure of TDKC algorithm is outlined, followed by an introduction of its improvements. To evaluate the performance of the proposed algorithm and compare it with other well-known algorithms, the performance evaluation metrics and scoring system for algorithm comparison are further elaborated.

3.1. DP algorithm

DP algorithm is notable for the ease of implementation. It effectively reduces the number of points in a trajectory while maintaining its overall shape, which simplifies data processing and visualization significantly (Bai et al., 2023). The algorithm adapts to different levels of detail by setting a tolerance parameter, providing adaptability and versatility across various applications (Xin et al., 2021). It also serves as a foundation for advanced techniques in trajectory data simplification (Tang et al., 2021).

In DP algorithm, a point is identified as a key point if it has an obvious deviation that exceeds a distance threshold. The algorithm continues execution on the sub-trajectories split by the identified key point until no further key point is found. If a trajectory is denoted by

$$T = \{p_1, p_2, \dots, p_i, \dots, p_{n-1}, p_n\} \tag{1}$$

where p_i is the i th trajectory point in T, and n is the number of points. p_i is a quintuple in the format of

$$p_i = (t_i, x_i, y_i, s_i, c_i) \tag{2}$$

with x_i and y_i representing position information, s_i and c_i representing speed and course information at timestamp t_i respectively. Then for an input trajectory T, the flowchart of DP algorithm is presented in Fig. 2.

Algorithmically, Fig. 2 is elaborated below.

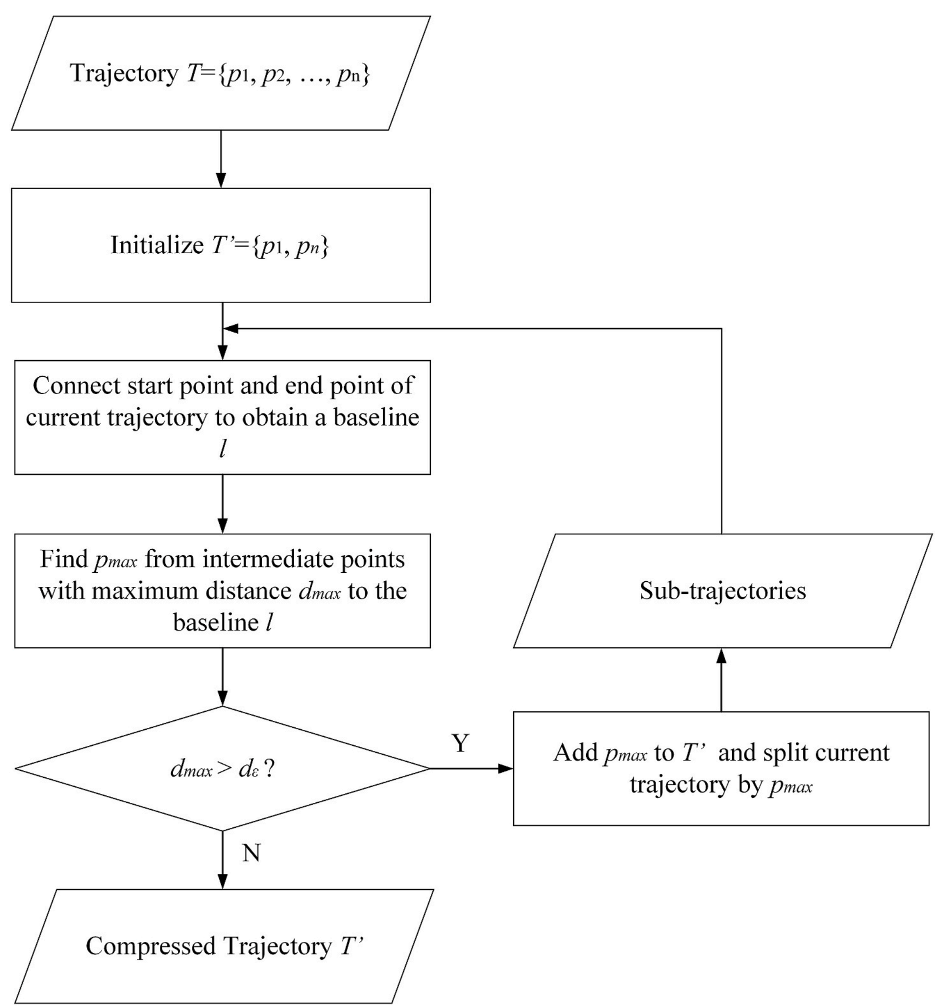


Fig. 2. DP algorithm procedure.

Step 1: Initiate compressed trajectory by $T' = \{p_1, p_n\}$. If T has more than two points, go to Step 2, otherwise, the algorithm ends.
Step 2: Connect the start point p_s and end point p_e of current trajectory to obtain a baseline l.
Step 3: For each intermediate point p_i, calculate its distance d_i to the baseline l, and denote the maximum distance by $d_{\max}$, the relevant point is then marked by $p_{\max}$.
Step 4: When $d_{\max}$ is greater than the pre-defined threshold d_ε, add p_{max} to T', and split current trajectory into two sub-trajectories by p_{max}. Otherwise, the process ends.
Step 5: For each sub-trajectory re-execute Step 2 to Step 4 until the algorithm finally ends. Then T' is the compressed trajectory of T.

The pseudocode of DP algorithm is presented in Algorithm 1. An example in Fig. 3 depicts how DP algorithm compresses a trajectory $T = \{p_1, p_2, p_3, p_4, p_5, p_6, p_7, p_8\}$ to $T' = \{p_1, p_3, p_6, p_8\}$. Fig. 3(a) presents the entire trajectory, the algorithm first connects p_1 and p_8 to calculate distance of all intermediate points to the baseline $\overline{p_1 p_8}$, where p_3 is found to have the maximum distance d_{max} and marked as the key point as depicted in Fig. 3(b). Then split by p_3, the algorithm continues to calculate the d_{max} in sub-trajectories. In Fig. 3(c), d_{max} is found related to p_2 in sub-trajectory $SubT_1 = \{p_1, p_2, p_3\}$, However, since d_{max} in $SubT_1$ is smaller than the preset threshold d_ε, the process in $SubT_1$ halts. Meanwhile, in $SubT_2 = \{p_3, p_4, p_5, p_6, p_7, p_8\}$, p_6 is identified as the key point. Because no additional key points exist in sub-trajectories of $SubT_2$, the process stops. Finally, by removing all non-key points, the compressed trajectory T' is obtained and shown in Fig. 3(d).

3.2. TDKC algorithm

Top-Down Kinematic Compression (TDKC) algorithm is an advanced version of DP algorithm. It applies Synchronous Euclidean Distance (SED) and Synchronous Velocity Difference (SVD) to incorporate kinematic attributes including time, longitude, latitude, speed, and course within AIS data for trajectory compression. In addition, TDKC algorithm employs a Compression Binary Tree (CBT) to address the recursion termination problem commonly encountered in top-down algorithms. An adaptive strategy is further employed to determine the compression thresholds automatically. The algorithm compresses a trajectory through three steps.

Step 1: Construct a CBT by using SED and SVD as the information measurement methods.
Step 2: Adaptively set the thresholds by calculating mean SED and SVD values from the CBT.
Step 3: Identify the key nodes in the CBT and preserve the point in each key node to obtain the compressed trajectory.

An overview of TDKC algorithm's procedure is presented in Fig. 4. Details will be elaborated in the following sections.

3.2.1. Information difference measurement

I. Spatial Distance

Spatial distance is a fundamental factor in trajectory compression algorithms as it quantifies the physical separation between consecutive data points in a trajectory (Li et al., 2016). In DP algorithm, the compression can be understood as there exists an alternative point on the baseline to represent the deleted point when the distance is inconspicuous and acceptable. To assess such deviation, DP algorithm adopts the most commonly used Perpendicular Euclidean Distance (PED). In Fig. 5, a segment within a trajectory in Cartesian coordinate system contains p_i, p_{i+1}, and p_{i+2}. Denote the vertical projection point of p_{i+1} onto the baseline $\overline{p_i p_{i+2}}$ by p_{i+1}', the distance between p_{i+1} and p_{i+1}' is the PED of p_{i+1} calculated by

$$ped_{i+1} = \sqrt{(x_{i+1} - x_{i+1}')^2 + (y_{i+1} - y_{i+1}')^2} \tag{3}$$

Since only positional information is adopted in the calculation of PED, it manifests a conspicuous limitation in effectively taking advantage of the temporal information inherent in trajectory data derived from Global Positioning System (GPS). This deficiency has been improved by the introduction of Synchronous Euclidean Distance (SED) in Top-Down Time-Ratio (TD-TR) algorithm, which assumes the discarded point can be reconstructed by linear interpolation and has been proved to demonstrate superior performance on temporally annotated trajectory data (Meratnia and de By, 2004). Suppose there is a point p_{i+1}'' on the baseline $\overline{p_i p_{i+2}}$ that satisfies

$$t_{i+1}'' = t_{i+1} \tag{4}$$

$$x_{i+1}'' = x_i + \frac{t_{i+1} - t_i}{t_{i+2} - t_i}(x_{i+2} - x_i) \tag{5}$$

$$y_{i+1}'' = y_i + \frac{t_{i+1} - t_i}{t_{i+2} - t_i}(y_{i+2} - y_i) \tag{6}$$

then the SED of p_{i+1} is determined by

$$sed_{i+1} = \sqrt{(x_{i+1} - x_{i+1}'')^2 + (y_{i+1} - y_{i+1}'')^2} \tag{7}$$

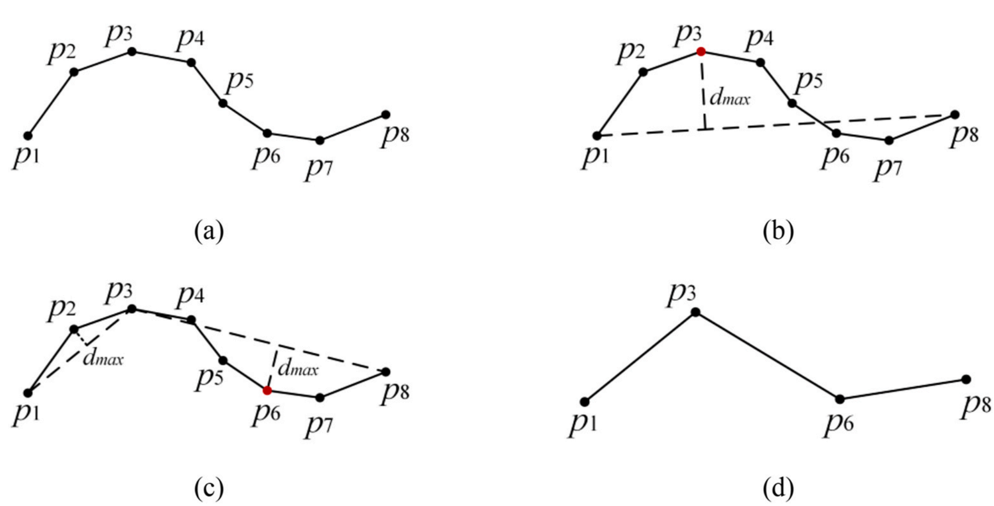


Fig. 3. An example of DP algorithm.

Algorithm 1. DP algorithm.

```
Function: DP(traj, threshold)
Input:    traj: The trajectory to be processed.
          threshold: Threshold for identifying split point.
Output:   cp_traj: Compressed trajectory.
if len(traj.pts) <= 2:
    return traj
end if
start_pt = traj.pts[0]
end_pt = traj.pts[-1]
max_d = NEG_INFINITY
index = -1
for pt in traj.pts[1:-1]:
    d = CalculateDist(start_pt, end_pt, pt)
    if d > max_d:
        max_d = d
        index = traj.pts.index(pt)
    end if
end for
if max_d > threshold:
    left = DP(Trajectory(traj.pts[0:index + 1]), traj.info), threshold)
    right = DP(Trajectory(traj.pts[index:]), traj.info), threshold)
    cp_traj = Trajectory(left.pts + right.pts[1:], traj.info)
else:
    cp_traj = Trajectory([start_pt, end_pt], traj.info)
end if
return cp_traj
```

It should be highlighted that the calculation of PED and SED under Cartesian coordinate system neglects the sphere shape of the Earth and distorts the trajectory significantly as latitude moves farther from the equator (Karney, 2013). In Fig. 6 an example shows how the calculation in Cartesian coordinate system introduces the distortion.

The acceptance of distortion is predicated on the premise of calculating the distance on a small scale, yet the specific definition of a small scale remains vague (Ma et al., 2022). For the purpose of mitigating the introduction of significant errors, all trajectories in this study are maintained in the geographic coordinate system. Based on spherical trigonometry, Haversine formula provides an accurate method to calculate distances between two points on the Earth's surface (Sinnott, 1984). Therefore, we employ Haversine formula for spatial distance calculations. Consider two points p_i and p_j, the distance between them is calculated by

$$d_{i,j} = 2r\,\arcsin\left(\sqrt{\sin^2\left(\frac{\Delta lat_{i,j}}{2}\right) + \cos(lat_i)\cos(lat_j)\sin^2\left(\frac{\Delta lon_{i,j}}{2}\right)}\right) \tag{8}$$

Here, $d_{i,j}$ represents the distance between the two points, r is the radius of the Earth in meters, $\Delta lon_{i,j}$ and $\Delta lat_{i,j}$ are the differences in latitude and longitude between the two points, calculated by

$$\Delta lon_{i,j} = lon_j - lon_i \tag{9}$$

$$\Delta lat_{i,j} = lat_j - lat_i \tag{10}$$

And lon_i, lon_j, lat_i, and lat_j represent the longitudes and latitudes of the two points.

II. Velocity Difference

Velocity difference is a supplementing factor that helps preserve points with significant speed and course change (Liu et al., 2023). In some studies, treating speed and course separately undeniably increases the number of thresholds to be determined for key point identification (Zhou et al., 2023b). Moreover, the disassociation may intensify the difficulty of identifying truly important key points since the dimensions and scales are different in these two attributes. In contrast, analyzing the velocity vector offers a complete description of motion characteristics, allowing for a more accurate and intuitive assessment of differences in a unified manner. To illustrate, if one defines two velocity vectors that exhibit speed and course differences greater than 1 knot and 5° respectively as having a significant velocity difference, then vectors shown in Fig. 7 will be identified as non-significant. However, if we consider the vectors directly and still apply a threshold of 1 knot, they then demonstrate a significant velocity difference, thereby leading to a more meaningful and accurate measurement. Hence, we use velocity vector to consider ship's speed and course simultaneously and introduce a new velocity difference measurement method in this study.

For a trajectory point p_i, the velocity can be represented by

$$\boldsymbol{v}_i = (vx_i, vy_i) \tag{11}$$

where vx_i and vy_i denote the speed that causing changes in longitude and latitude respectively. The value of vx_i and vy_i is determined by the decomposition of SOG using COG through the following equations

$$vx_i = s_i \sin c_i \tag{12}$$

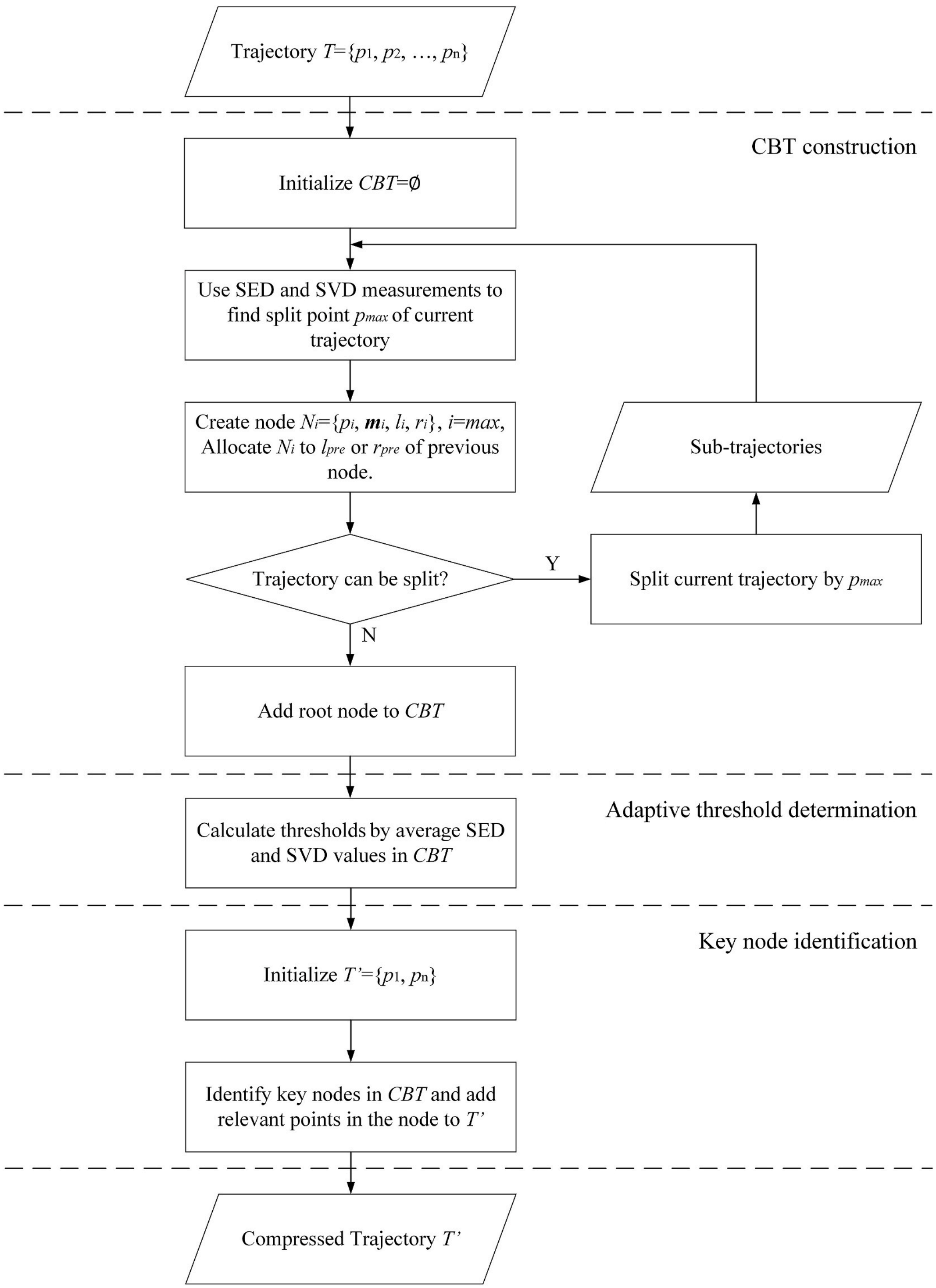


Fig. 4. TDKC algorithm procedure.

$$vy_i = s_i \cos c_i \tag{13}$$

Then the velocity difference between p_i and p_j can be calculated by

$$\Delta \boldsymbol{v}_{i,j} = \boldsymbol{v}_j - \boldsymbol{v}_i \tag{14}$$

In order to capitalize on temporal information, the concept of Synchronous Velocity Difference (SVD) is proposed in this paper, which is a novel contribution, compared to the other top-down algorithms. For the segment in Fig. 5, first the speed change $\Delta s_{i,i+2}$ and course change $\Delta c_{i,i+2}$ are assessed through

$$\Delta s_{i,i+2} = s_{i+2} - s_i \tag{15}$$

$$\Delta c_{i,i+2} = ((c_{i+2} - c_i + 180) \mod 360) - 180 \tag{16}$$

Since the course change can be either clockwise or anticlockwise, its calculation in Eq. (16) follows the rule of shortest course change in the consideration of a gradually decreasing time scale during the compression. Then, based on the assumption of linear change, the estimated speed and course of p_{i+1}'' are computed by

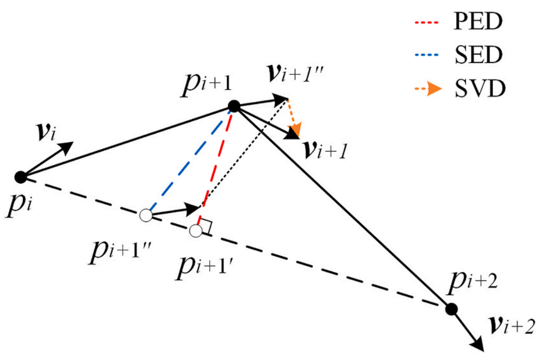


Fig. 5. PED, SED, and SVD

$$s_{i+1}{}'' = s_i + \frac{t_{i+1} - t_i}{t_{i+2} - t_i} \Delta s_{i,i+2} \tag{17}$$

$$c_{i+1}{}'' = c_i + \frac{t_{i+1} - t_i}{t_{i+2} - t_i} \Delta c_{i,i+2} \tag{18}$$

Finally, for a more intuitive representation, we take the magnitude of $\Delta \boldsymbol{v}_{i+1,i+1''}$ to represent the SVD of p_{i+1}, which is derived from

$$svd_{i+1} = \sqrt{(vx_{i+1}{}'' - vx_{i+1})^2 + (vy_{i+1}{}'' - vy_{i+1})^2} \tag{19}$$

III. Aggregated Measurement and Split Point Determination

TDKC algorithm applies an aggregated measurement method that integrates SED and SVD to consider kinematic attributes simultaneously. Define sets S_{SED} and S_{SVD} to represent SED and SVD values of all intermediate points

$$S_{SED} = \{sed_i | start < i < end\} \tag{20}$$

$$S_{SVD} = \{svd_i | start < i < end\} \tag{21}$$

where *start* and *end* are the indexes of current trajectory's start point and end point in a recursion step. The information difference measurement of p_i is a two-dimensional vector, which is denoted by $\boldsymbol{m}_i = (sed_i, svd_i)$. Then, to incorporate SED and SVD measurements, z-score normalization is adopted to mitigate the influence of varying attribute scales and units:

$$sed_i^{norm} = \frac{sed_i - \mu_{S_{SED}}}{\sigma_{S_{SED}}} \tag{22}$$

$$svd_i^{norm} = \frac{svd_i - \mu_{S_{SVD}}}{\sigma_{S_{SVD}}} \tag{23}$$

Here, $\mu_{S_{SED}}$, $\mu_{S_{SVD}}$, $\sigma_{S_{SED}}$, and $\sigma_{S_{SVD}}$ represent the mean and standard deviation of S_{SED} and S_{SVD} respectively. The normalized aggregated measurement $\boldsymbol{m}_i$ is represented by $\boldsymbol{m}_i^{norm} = (sed_i^{norm}, svd_i^{norm})$. As shown in Fig. 8, z-score transformation shifts the distribution of $\boldsymbol{m}_i$ towards the center of the origin. Since we consider sed_i^{norm} and svd_i^{norm} to reflect equal importance in information difference measurement, we define the magnitude of $\boldsymbol{m}_i^{norm}$ by

$$|\boldsymbol{m}_i^{norm}| = sed_i^{norm} + svd_i^{norm} \tag{24}$$

The example in Fig. 8(b) demonstrates that a higher $|\boldsymbol{m}_i^{norm}|$ signifies a point either concurrently possesses larger SED and SVD values, or exhibits one of them with an exceptionally high value compared to other points within this recursion. Therefore, the split point p_{max} in each recursion of TDKC is the point with the maximum normalized aggregated measurement value denoted by $|\boldsymbol{m}_{max}^{norm}|$. The pseudocode of the process to determine the split point in a recursion is illustrated in Algorithm 2.

3.2.2. Compression Binary Tree

The recursive nature of top-down algorithms inevitably results in the recursion termination problem. The problem determines that when the terminating condition is met, the algorithm will stop splitting current trajectory without examining the sub-trajectory in next recursion, even if the point with maximum distance still satisfies the criteria for being identified as a key point. Suppose there is a trajectory $T = \{p_1, p_2, p_3, p_4, p_5, p_6, p_7\}$ in Fig. 9, in the first recursion, p_5 is found the point with the maximum distance along the trajectory. If $d_5 > d_\varepsilon$, T will be split into $subT_1 = \{p_1, p_2, p_3, p_4, p_5\}$ and $subT_2 = \{p_5, p_6, p_7\}$. Then in the subsequent recursion process for $subT_1$, p_3 is identified to be preserved with the maximum distance d_3. However, if $d_5 \leqslant d_\varepsilon$, the process will terminate immediately in the first recursion, regardless of whether $d_3 > d_\varepsilon$ or not. In this situation, p_3 will never be preserved though $d_3 > d_\varepsilon$.

As a top-down compression process can be depicted in a tree structure (Zhai et al., 2017), we introduce the concept of Compression Binary Tree (CBT) in TDKC algorithm to address the problem. CBT is a data structure that stores all the key points and their relevant measurements along the recursion process. The construction of CBT necessitates a zero-threshold setting to exhaust the recursion process. Denoted by *CBT*, if it is a non-empty tree, the structure can be defined as follows:

$$CBT = \{Root\} \tag{25}$$

where *Root* is the root node of *CBT* acquired from the first level of

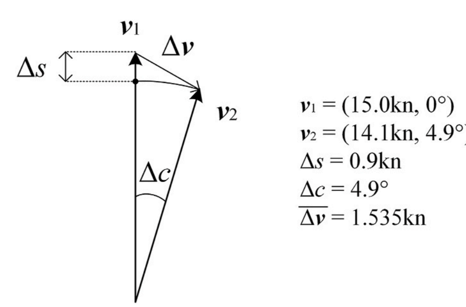
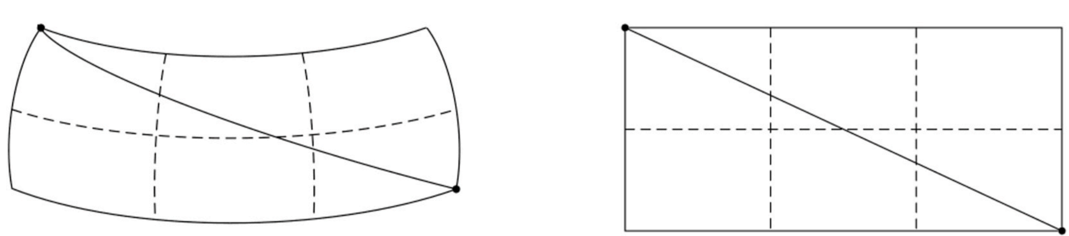
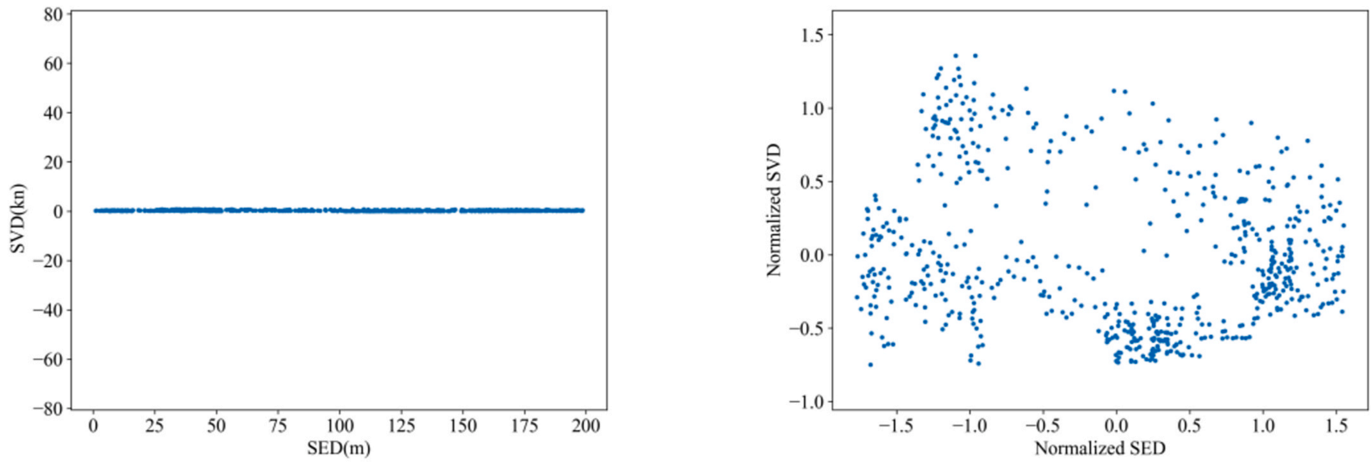


Fig. 7. An example of identifying significant velocity difference.



(a) Geographic coordinate system (b) Cartesian coordinate system

Fig. 6. Trajectory distortion.



(a) Original distribution.   (b) Transformed distribution.

Fig. 8. Example of z-score normalization.

Algorithm 2. Split point identification.

```
Function: FindSplitPt(pts)
Input:    pts: Points in the trajectory to be processed.
Output:   split_pt: The identified split point.
          ms_vector: The relevant information difference measurement results of the split point.
if len(pts) <= 2:
    return None
end if
start_pt = pts[0]
end_pt = pts[-1]
initialize sed_set, svd_set
for pt in pts[1:-1]:
    sed = GetSED(start_pt, end_pt, pt)
    svd = GetSVD(start_pt, end_pt, pt)
    sed_set.append(sed)
    svd_set.append(svd)
end for
norm_sed_set = ZScoreNormalize(sed_set)
norm_svd_set = ZScoreNormalize(svd_set)
initialize norm_aggregated
for norm_sed, norm_svd in norm_sed_set, norm_svd_set:
    norm_aggregated.append(norm_sed + norm_svd)
end for
max_index = norm_aggregated.index(max(norm_aggregated))
split_pt = pts[max_index + 1]
ms_vector = [sed_set[max_index], svd_set[max_index]]
return split_pt, ms_vector
```

recursion. In each recursion, a node N_i will be created. It is structured as follows:

$$N_i = \{p_i, \boldsymbol{m}_i, l_i, r_i\} \tag{26}$$

Here, i indicates the order of the point, p_i is the split point identified with maximum aggregated measurement value in this recursion, and $\boldsymbol{m}_i$ is the relevant measurement vector. p_i will split current trajectory into two sub-trajectories. Then, l_i and r_i store the nodes created in the next level of recursion respectively. The CBT construction algorithm is detailed in Algorithm 3.

3.2.3. Adaptive threshold determination and key node identification

In the process of CBT construction, the measurement of information difference gradually transitions from a macro-scale to a micro-scale. As recursion depth grows, the frequency of measurement at each depth also

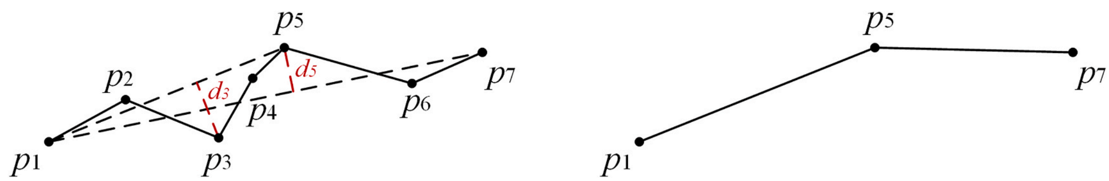


Fig. 9. Recursion termination problem.

Algorithm 3. CBT construction.

```
Function: ConstructCBT(pts)
Input:    pts: Points in the trajectory to be processed.
Output:   node: A CBT's node.
if len(pts) <= 2:
        return None
end if
start_pt = pts[0]
end_pt = pts[-1]
split_pt, ms_vector = FindSplitPt(pts)
l = ConstructCBT(pts[0:pts.index(split_pt) + 1])
r = ConstructCBT(pts[pts.index(split_pt):])
node = [split_pt, ms_vector, l, r]
return node
```

increases. Consequently, it can be inferred that the measured values of information difference tend to decrease and eventually stabilize, which has also been observed by other researchers (Tang et al., 2021). Considering the scale of the entire trajectory, the information difference at the micro-scale can be deemed negligible. Therefore, the average value of S_{SED} and S_{SVD} is utilized as adaptive thresholds, denoted by

$$sed_{\varepsilon} = \frac{1}{n-2}\sum_{2}^{n-1} sed_i \tag{27}$$

$$svd_{\varepsilon} = \frac{1}{n-2}\sum_{2}^{n-1} svd_i \tag{28}$$

Since a CBT contains all points in the trajectory apart from the first and last points, we introduce the concept of key node to preserve points. First, p_i is identified as a key point if $\boldsymbol{m}_i$ satisfies

$$(sed_i > sed_{\varepsilon}) \vee (svd_i > svd_{\varepsilon}) \tag{29}$$

Here, sed_{ε} and svd_{ε} are thresholds for SED and SVD measurements respectively. Then, a key node is defined as a node that either directly contains a key point or possesses at least one child node with a key point. All points residing in the key nodes are designated for retention. An example is shown in Fig. 10. Suppose $\boldsymbol{m}_4$ satisfies Eq. (29) while $\boldsymbol{m}_3$ does not, the recursion will stop at N_3 in traditional algorithms, resulting in the exclusion of p_3 and p_4. However, since N_3 and N_4 both satisfy the definition of key node, p_3 and p_4 will be retained in TDKC. Hence, a key node identification algorithm is presented to further solve the recursion termination problem in Algorithm 4. In conclusion, the pseudocode of TDKC algorithm is presented in Algorithm 5.

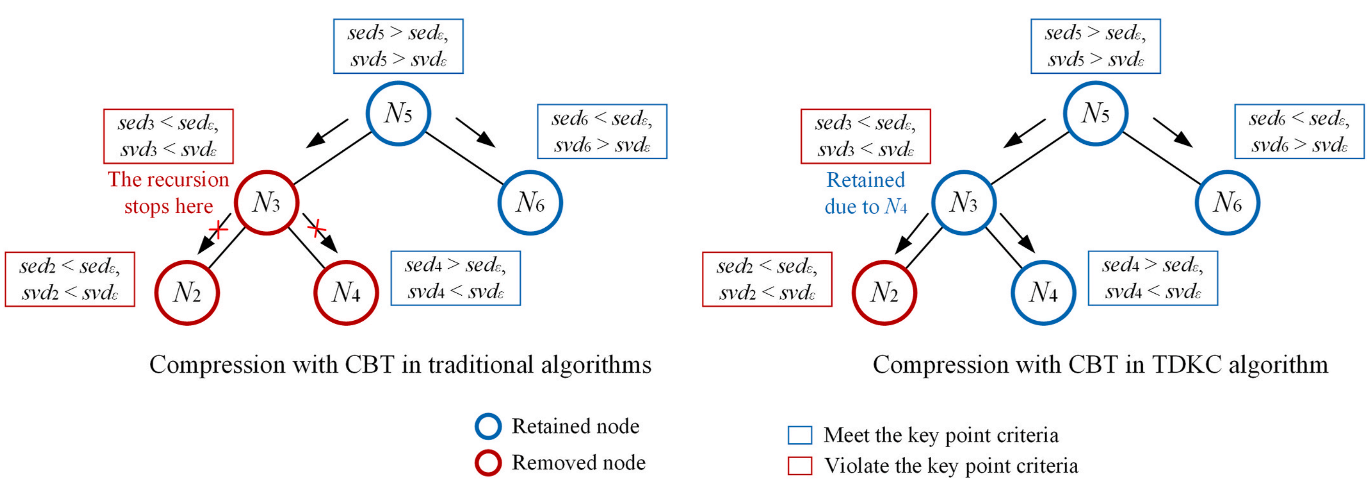


Fig. 10. An example of CBT.

Algorithm 4. Key node identification.

Function: IdentifyKeyNode(*node*, *threshold_sed*, *threshold_svd*, *key_nodes*)

Input: *node*: A CBT's node.
threshold_sed: SED threshold.
threshold_svd: SVD threshold.
key_nodes: Key nodes set updated during the recursion.

Output: Boolean.

```
if node is not None:
    if node.ms_vector[0] > threshold_sed or node.ms_vector[1] > threshold_svd:
        key_nodes.append(node)
        return True
    else:
        l = IdentifyKeyNode(node.l, threshold_sed, threshold_svd, key_nodes)
        r = IdentifyKeyNode(node.r, threshold_sed, threshold_svd, key_nodes)
        if l or r:
            key_nodes.append(node)
            return True
        else:
            return False
        end if
    end if
else:
    return False
end if
```

3.3. *Performance evaluation metrics*

The performance of a compression algorithm is usually evaluated by a variety of metrics, such as running time (RT), compression rate (CR), length loss rate (LLR), and distance-based similarity (SMD) metrics in previous studies (Amigo et al., 2021). RT is a general metric to assess the efficiency of the algorithm by calculating the duration of the execution. Others are often employed to assess the effectiveness of the compression. In addition, a novel velocity-based similarity (SMV) metric is proposed to evaluate the performance from the perspective of velocity. The metrics for compression effectiveness are specified in more detail in the following sections.

3.3.1. *Compression rate*

CR is defined as the ratio between the number of eliminated points and total points. Individual compression rate (ICR) is identified as the ratio between the number of discarded points k and total points n of an individual trajectory, calculated by

$$ICR = \frac{k}{n} \tag{30}$$

Overall compression rate (OCR) is the metric to evaluate compression rate within the entire dataset that contains q trajectories, characterized by

$$OCR = \frac{\sum_{i=1}^{q} k_i}{\sum_{i=1}^{q} n_i} \tag{31}$$

3.3.2. *Length loss rate*

LLR evaluates the effectiveness of a compression algorithm by measuring the difference between the length of compressed trajectory and original trajectory. The length of a trajectory is computed by

$$L = \sum_{i=1,j=i+1}^{n-1} d_{i,j} \tag{32}$$

where $d_{i,j}$ is the distance between p_i and p_j. Denote the length of the compressed trajectory by L', the individual length loss rate (ILLR) is

$$ILLR = \frac{L - L'}{L} \tag{33}$$

Similarly, overall length loss rate (OLLR) is further determined by

$$OLLR = \frac{\sum_{i=1}^{q} (L_i - L_i{}')}{\sum_{i=1}^{q} L_i} \tag{34}$$

when the dataset comprises m trajectories.

3.3.3. *Similarity metrics*

As for similarity metrics, Dynamic Time Wrapping (DTW) algorithm is a useful tool to gauge the similarity between the original and compressed trajectories (Toohey and Duckham, 2015). Suppose there are two trajectories T_1 and T_2, the SMD can be measured by

$$SMD(T_1, T_2) = D(n_1, n_2) \tag{35}$$

Here, n_1 and n_2 are the numbers of points in T_1 and T_2 respectively. For point p_i in T_1 and point p_j in T_2, cumulative distance $D(i,j)$ is defined as

$$D(i,j) = d_{i,j} + \min\{D(i,j-1), D(i-1,j), D(i-1,j-1)\} \tag{36}$$

The boundary conditions of Eq. (36) are $D(1,1)$ and $D(m,n)$. However, conventional similarity measurement neglects the velocity information. As a supplement, we put forward a SMV measurement depicted as follows

Algorithm 5. TDKC algorithm.

Function: TDKC(*traj*)
Input: *traj*: The trajectory to be processed.
Output: *cp_traj*: Compressed trajectory.

```
# Construct CBT
cbt = ConstructCBT(traj.pts)

# Calculate thresholds
sed_set, svd_set = GetMeasurement(cbt)
threshold_sed = Mean(sed_set)
threshold_svd = Mean(svd_set)

# Identify key nodes
initialize key_nodes
IdentifyKeyNode(cbt, threshold_sed, threshold_svd, key_nodes)

# Preserve points
initialize preserved_pts
preserved_pts.append(traj.pts[0])
for node in key_nodes:
    if node.split_pt is not in preserved_pts:
        preserved_pts.append(node.split_pt)
    end if
end for
preserved_pts.append(traj.pts[-1])
cp_traj = Trajectory(preserved_pts, traj.info)
return cp_traj
```

$$SMV(T_1, T_2) = V(n_1, n_2) \tag{37}$$

where cumulative velocity difference $V(i,j)$ is calculated by

$$V(i,j) = |\Delta \boldsymbol{v}_{i,j}| + \min\{V(i,j-1), V(i-1,j), V(i-1,j-1)\} \tag{38}$$

Considering different trajectories may manifest various compression rates, cumulative distance and velocity difference can experience deviations due to the differing number of points before and after compression. Thus, we adopt the normalized version of SMD and SMV, which are employed to render them comparable among time series of various lengths (Leodolter et al., 2021).

$$SMD_{norm}(T_1, T_2) = \frac{SMD(T_1, T_2)}{n_1 + n_2} \tag{39}$$

$$SMV_{norm}(T_1, T_2) = \frac{SMV(T_1, T_2)}{n_1 + n_2} \tag{40}$$

3.4. Scoring system for algorithms comparison

Each metric is used as input to a well-designed scoring system, which is employed to illustrate the strengths and limitations of the algorithms implemented during comparative analysis. The scoring system is relative since we only compare the performance superiority among these algorithms. Hence, the quantification from metrics to scores necessitates a normalization process (Hwang and Yoon, 1981). In our proposed scoring system, each metric is scored on a scale from 1 to 5, representing from very poor (1) to very good performance (5). For RT, LLR, SMD, and SMV performance metrics, lower values indicate superior performance. Conversely, a higher CR value signifies better performance. Therefore, the quantification contains two basic scoring functions,

$$score_{low}(\alpha_i) = 5 - 4 \cdot \frac{\alpha_i - \min(M)}{\max(M) - \min(M)} \tag{41}$$

$$score_{high}(\alpha_i) = 1 + 4 \cdot \frac{\alpha_i - \min(M)}{\max(M) - \min(M)} \tag{42}$$

where $score_{low}$ and $score_{high}$ are used for negative and positive metrics respectively. In the functions, $\alpha_i \in M$ is the input metric value of i th algorithm for quantification, where $M = \{\alpha_1, \alpha_2, ..., \alpha_\tau\}$ denotes the metric value set for all algorithms and τ is the number of the algorithms used for study. $\min(M)$ and $\max(M)$ represent the minimum and maximum values in set M. The quantification of different metrics varies depending on the consideration of individual trajectory and overall trajectory evaluations. The detailed quantification is elaborated as follows.

I. RT

It is known that a longer trajectory sequence length necessitates more processing time for all algorithms. In practice, the trajectory sequence lengths within the entire water area are varied. The total running time might be affected by the lengthiest trajectory sequence significantly, which hides the evaluation on processing trajectory with short sequences. Therefore, we adopt a weighted quantification strategy for RT. Suppose the trajectories are divided into λ length types by analyzing the trajectory length distribution in the water area. The quantification of RT related to i th algorithm is then determined by

$$QRT_i = \sum_{j=1}^{\lambda} w_j^{RT} \cdot score_{low}(RT_j) \tag{43}$$

The weight w_j^{RT} is determined by the ratio between the number of trajectories in j th length type and the total number of all trajectories.

II. CR and LLR

Given that both CR and LLR contain individual and overall trajectory evaluations, the quantifications of CR and LLR are considered from three perspectives.

$$QCR_i = w_1^{CR} \cdot score_{high}(OCR_i) + w_2^{CR} \cdot score_{high}(\mu_{ICR_i}) + w_3^{CR} \cdot score_{low}(\delta_{ICR_i}) \tag{44}$$

$$QLLR_i = w_1^{LLR} \cdot score_{low}(OLLR_i) + w_2^{LLR} \cdot score_{low}(\mu_{ILLR_i}) + w_3^{LLR} \cdot score_{low}(\delta_{ILLR_i}) \tag{45}$$

In Eqs. (44) and (45), $score_{high}(OCR_i)$ and $score_{low}(OLLR_i)$ determine the CR score and LLR score on overall trajectories. Subsequently, $score_{high}(\mu_{ICR_i})$ and $score_{low}(\mu_{ILLR_i})$ use the average ICR and LLR values of all trajectories to compute the scores. Finally, $score_{low}(\delta_{ICR_i})$ and $score_{low}(\delta_{ILLR_i})$ calculate the scores by considering the Interquartile Ranges (IQRs), which represent the range between the first quartile (Q1) and the third quartile (Q3) of ICR and ILLR sets. Taking QCR_i for example, $score_{high}(OCR_i)$ evaluates the CR scores in terms of the overall trajectories, while $score_{high}(\mu_{ICR_i})$ and $score_{low}(\delta_{ICR_i})$ emphasis on the distribution of individual trajectory performance. Therefore, the values of w_1^{CR}, w_2^{CR}, and w_3^{CR} are set to 0.5, 0.25, and 0.25 respectively. The same applies for w_1^{LLR}, w_2^{LLR}, and w_3^{LLR} .

III. SMD and SMV

As there is no difference between using total or mean values to compare algorithmic performances through SMD and SMV, we adopt mean and IQR scoring for quantifying SMD and SMV.

$$QSMD_i = w_1^{SMD} \cdot score_{low}(\mu_{SMD_i}) + w_2^{SMD} \cdot score_{low}(\delta_{SMD_i}) \tag{46}$$

$$QSMV_i = w_1^{SMV} \cdot score_{low}(\mu_{SMV_i}) + w_2^{SMV} \cdot score_{low}(\delta_{SMV_i}) \tag{47}$$

Here, since mean and IQR scoring both consider the distribution of individual trajectory performance, they equally contribute to the quantification. Hence, w_1^{SMD} and w_2^{SMD} are both set to 0.5, and likewise for w_1^{SMV} and w_2^{SMV}.

4. Case study

In this section, a case study is presented to validate the proposed algorithm. All experiments were conducted using Python 3.8 on a computer equipped with a 12th Gen Intel(R) Core(TM) i7-12700KF processor and 32 GB of RAM, running a 64-bit Windows 10 operating system. Additionally, the algorithms chosen for comparison are introduced, accompanied by the reasons for their selection and relevant parameter settings.

4.1. Experiment data

4.1.1. Data preprocessing

The AIS data for this study was collected from Gulf of Finland, with longitude ranging from 19 °E to 31 °E and latitude ranging from 59 °N to 61 °N. The time duration spanned from 01:00:00 on 11th to 00:59:59 on September 18, 2021. Considering that raw AIS data inevitably contains errors such as invalid MMSI numbers and kinematic information due to the factors related to transmission environment, equipment, and human input, a preprocessing procedure is required to remove these anomalies (Guo et al., 2021b). Therefore, we first identified and removed those AIS data with obviously incorrect information that conflicts with AIS message definitions (ITU, 2014). Then, the original trajectory dataset was constructed based on MMSI and the chronological order of timestamps. Finally, to further enhance the data quality, a clustering-based anomaly detection method proposed by Guo et al. (2021a) was used to eliminate hidden anomalies. Consequently, an example of the preprocessed AIS trajectory data is shown in Table 2.

After preprocessing, the dataset contained 18112237 trajectory points. Since the comparative investigation of algorithms requires ship size information, we excluded trajectories without valid ship size information. In addition, considering that too short trajectories exhibit limited information, we assumed that trajectories within less than 10 points should remain unchanged for the purpose of preserving the integrity and meaningfulness of shorter trajectories. Consequently, 18083642 trajectory points within 4285 trajectories were kept, representing 99.8% retention for the sake of the study. The trajectories are depicted in Fig. 11.

4.1.2. Individual trajectory

To analyze the performance of the proposed algorithm comprehensively, an experiment was conducted on a randomly picked trajectory to initially reveal the algorithm's compression ability. The information of this trajectory is illustrated in Table 3. As portrayed in Fig. 12, the vessel was engaged in activities in the middle of Gulf of Finland, then returning to Paldiski South Harbor, and finally mooring at Port of Miiduranna.

4.1.3. Traffic areas

Then to ensure a more comprehensive comparative assessment and for verification purposes, another experiment focusing on various traffic areas was carried out. Ship navigation in open water areas tends to exhibit a tendency towards stability, characterized by infrequent and smooth changes in both speed and course. Conversely, in offshore and island regions, ships often experience frequent adjustments influenced by limited operation spaces (Zhou et al., 2023a). Hence, the dataset was divided into five subsets based on their locations and distinct geographic conditions to reveal the performance of the proposed algorithm in different water areas. Specifically, as depicted in Fig. 11, Area 1 represents the Finnish coast, which contains 3703621 points (20.4% of the total). This can be treated as a separate area in the view of multiple small islands close to the coast. Area 2 corresponds to the Estonian coast, consisting of 1378466 points (7.6% of the total). It is treated differently than Area 1, since the presence of the islands is significantly reduced. Area 3 denotes archipelago region, with 3173760 points distributed across them (17.5% of the total). Area 4 encompasses a combination of open water and island areas, where the open water area occupies the majority, comprising 3808307 points (21% of the total), so it can be treated as a mixture of Area 3 and 5. The remainder constitutes Area 5, characterized as an open water area, with 6019364 points within its boundaries (33% of the total). It should be noted that trajectories located predominately located at the borders with too few points were ignored (0.5% of the total). And algorithms were run individually in each area.

Table 2
An example of processed AIS trajectory data.

MMSI	Timestamp (s)	Longitude (°)	Latitude (°)	SOG (kn)	COG (°)
XXX136XXX	1631584513	25.017138	60.142292	7.2	133.0
XXX136XXX	1631584548	25.018911	60.141495	7.1	132.6
XXX136XXX	1631584574	25.020200	60.140915	7.1	131.5
XXX136XXX	1631584605	25.021698	60.140230	7.0	134.7
XXX136XXX	1631584631	25.022890	60.139620	7.1	133.4
XXX136XXX	1631584661	25.024382	60.138947	7.2	134.2

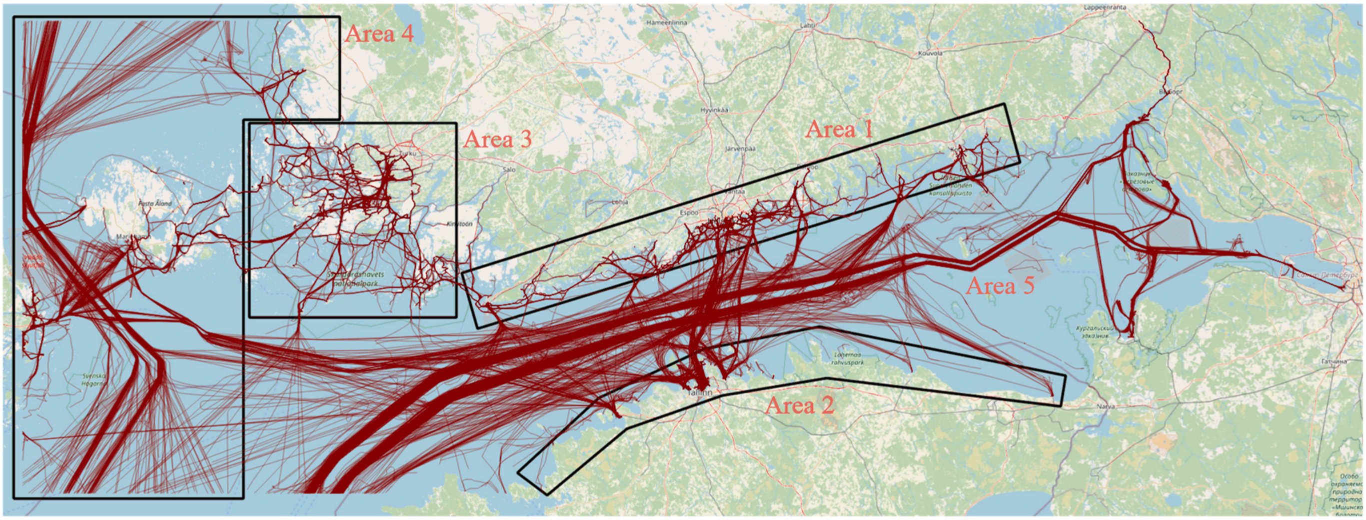


Fig. 11. Trajectory overview in Gulf of Finland.

Table 3
Information of selected trajectory.

Information	Details
MMSI	XXX035XXX
Start time	01:00:01 September 11, 2021 (UTC)
End time	05:18:01 September 13, 2021 (UTC)
Number of points	17823
Trajectory length	102.82 nm

4.2. Running time optimization

As an offline compression algorithm, TDKC algorithm shares the drawback of being time-consuming compared to online algorithms. The exhaustion of recursion process also necessitates additional execution time. In light of AIS specifications, position accuracy is generally maintained below 10 m under optimal conditions, ensuring reliable and precise tracking of ships (ITU, 2014). During the construction of CBT, as the recursion level increases, the time interval between successive points diminishes correspondingly, leading to a reduction in the distance between the points. Consequently, points separated by less than 10 m can be considered as occupying the same location. Furthermore, AIS messages are transmitted at specific time intervals based on the ship's navigation status as shown in Table 4. It can be observed that in most cases a navigating ship equipped with Class A transponder will transmit messages every 10 s. Albeit a Class B transponder permits a longer time interval, navigating less than 10 m in 3 min, which equals 0.108kn, can be another constraint that facilitates CBT construction. As such, not only can the processing speed of TDKC be accelerated, but the omission of data points with insignificant variations can also further optimize the compression efficiency.

A preliminary comparative study on Area 4 was conducted to help determine the strategy for using TDKC in subsequent experiments since Area 4 features a variety of trajectories in hybrid waters. In Table 5, TDKC-optimized refers to a running time optimized version of TDKC, while TDKC-original represents the original one. Apparently, TDKC-optimized demonstrates significant improvements in RT and OCR evaluations. Although its performance on OLLR, SMD, and SMV exhibits shortages compared to the original version, considering that in practice, processing time and compression efficiency are always the primary

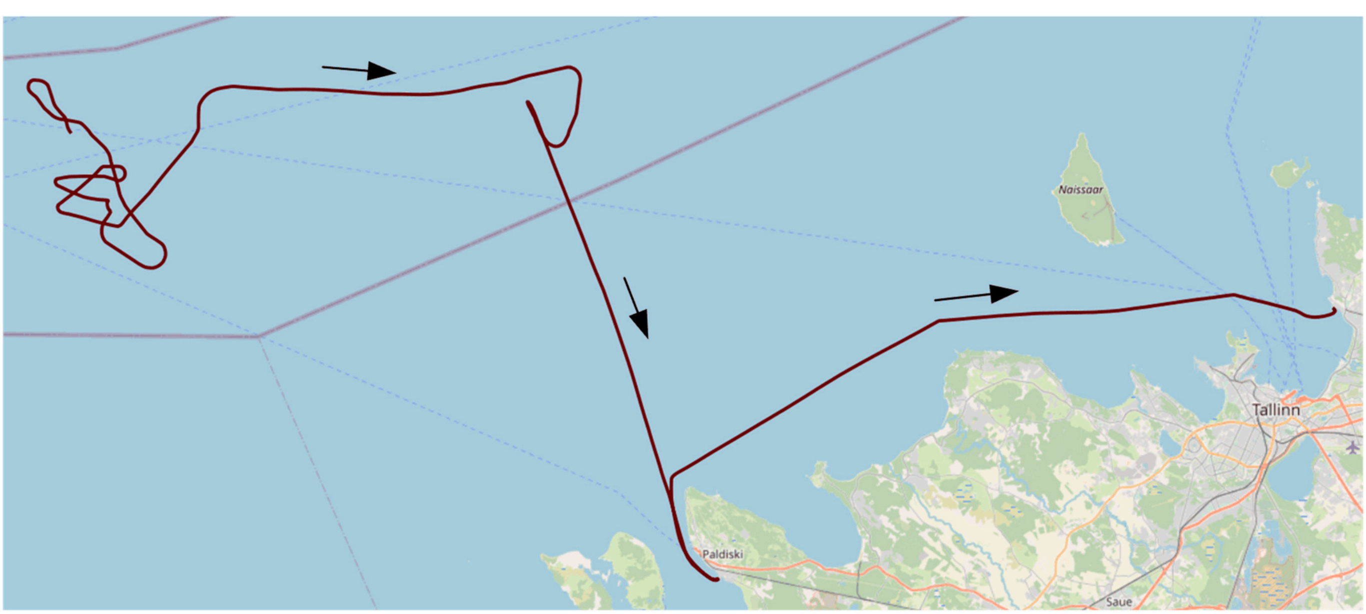


Fig. 12. Randomly selected trajectory for analysis.

Table 4
AIS message reporting intervals.

Class	Mode	Platform/Ship's Condition	Reporting Interval
A	SOTDMA	Ship at anchor or moored and moving slower than 3 knots	3 min
		Ship at anchor or moored and moving faster than 3 knots	10 s
		Ship 0–14 knots	10 s
		Ship 0–14 knots and changing course	3 1/3 s
		Ship 14–23 knots	6 s
		Ship 14–23 knots and changing course	2 s
		Ship >23 knots	2 s
		Ship >23 knots and changing course	2 s
B	SOTDMA	Shipborne mobile equipment moving slower than 2 knots	3 min
		Shipborne mobile equipment moving 2–14 knots	30 s
		Shipborne mobile equipment moving 14–23 knots	15 s
		Shipborne mobile equipment moving >23 knots	5 s
	CSTDMA	Shipborne mobile equipment moving slower than 2 knots	3 min
		Shipborne mobile equipment moving faster than 2 knots	30 s
		Search and rescue aircraft (airborne mobile equipment)	10 s
		Aids to navigation	3 min
		AIS base station	10 s

Table 5
Comparison of TDKC-original and TDKC-optimized algorithms on Area 4.

Metric	TDKC-original	TDKC-optimized
RT(s)	746.982	316.814
OCR	81.915%	93.635%
OLLR	0.764%	1.564%
SMD(m)	221.874	340.175
SMV(kn)	0.208	0.231

concerns, such a sacrifice is minor enough to be acceptable. Consequently, we applied a running time optimization strategy in TDKC for the subsequent experiments.

4.3. Algorithms selected for comparison

The presented algorithm was compared to other well-known algorithms and their improved versions. The algorithms and their characteristics selected for comparison are listed in Table 6, whilst justification for their selection is provided below.

DP algorithm is the basic top-down algorithm as illustrated in Section 3.1, which requires manual threshold configuration. Since most of advancing top-down algorithms stem from DP algorithm, it was reasonable to be included. TD-TR algorithm is an improved version of DP algorithm requiring manual threshold determination (Meratnia and de By, 2004). The incorporation of time series through SED resulted in its selection. In the experiments, the threshold for these two algorithms was set to 100 m, consistent with another improved DP (IDP) algorithm in (Liu et al., 2023). Besides position information, IDP algorithm also takes speed and course information into account, enhancing its effectiveness in preserving key trajectory features and rendering it a strong candidate for comparison. Thresholds of 0.1kn and 4° were used in the study as these are the recommended values by the authors for speed and course mutation points identification.

As proposed in (Zhang et al., 2016), an adaptive threshold of 0.8 times the ship length proves to be an effective adaptive strategy for DP algorithm enhancement. This algorithm has been selected for testing, designated as Adaptive-DP1 (ADP1) algorithm. Moreover, a novel

Table 6
Characteristics of the implemented algorithms.

Algorithm	Type of approach	Attribute in trajectories considered	Threshold determination	Literature
DP	Top-down	Position	Manual	Douglas and Peucker (1973)
TD-TR	Top-down	Time, position	Manual	Meratnia et al. (2004)
IDP	Top-down	Position, speed, course	Statistical	Liu et al. (2023)
ADP1	Top-down	Position	Adaptive	Zhang et al. (2016)
ADP2	Top-down	Position	Adaptive	Tang et al. (2021)
PDP	Top-down	Time, position, course	Statistical, manual	Zhao and Shi (2018)
SW	Sliding window	Time, position, course	Statistical	Gao and Shi (2019)
TDKC	Top-down	Time, position, speed, course	Adaptive	This paper

DP-based compression algorithm was presented by (Tang et al., 2021), which learns PED deviation change rate of trajectories in the water area and determines the threshold adaptively. It was also considered for the study and is denoted as Adaptive-DP2 (ADP2) algorithm.

Partition-DP (PDP) algorithm is another advanced version of DP algorithm presented by Zhao and Shi (2018), which concentrates on reducing the depth of recursion by considering the shape of trajectories. Since they suggested a range from 0.1 times and 10 times the ship length regarding position attribute, for the sake of comparison, we adopted 0.8 times as the threshold, which was the same as ADP1 algorithm. As for transition point identification, we followed their recommendation to set time interval to 50s and course change to 10° manually. Finally, we also selected a sliding window (SW) algorithm to represent online algorithm with reference to (Gao and Shi, 2019). SW takes both SED and course change into account. According to their study, the threshold of SED and course change was set to the ship width and 4.5° respectively.

5. Results and discussion

The case study introduced in the previous section contains experiments on an individual trajectory and different traffic areas respectively. The experimental results are presented in this section with a comprehensive discussion to reveal the superiority of the proposed algorithm.

5.1. Results of individual trajectory

5.1.1. Performance metric evaluation

The compression results for the illustrative trajectory are provided in Table 7 and Fig. 13. In terms of processing time, SW emerges as the fastest algorithm, requiring only 0.064s on the used computer. Following closely is PDP, consuming approximately 0.119s to compress the trajectory. DP, TD-TR, IDP, and ADP1 exhibit minimal differences in

Table 7
Compression results of different algorithms.

Algorithm	RT(s)	ICR	ILLR	SMD(m)	SMV(kn)
DP	0.718	99.551%	2.259%	637.550	0.228
TD-TR	0.836	99.551%	2.263%	637.194	0.228
IDP	0.730	61.611%	0.197%	106.086	0.057
ADP1	0.796	98.811%	1.757%	259.020	0.121
ADP2	1841.869	98.205%	1.692%	158.260	0.098
PDP	0.119	96.639%	1.511%	252.672	0.109
SW	0.064	62.543%	0.335%	374.142	0.088
TDKC	1.156	95.697%	1.607%	107.905	0.070

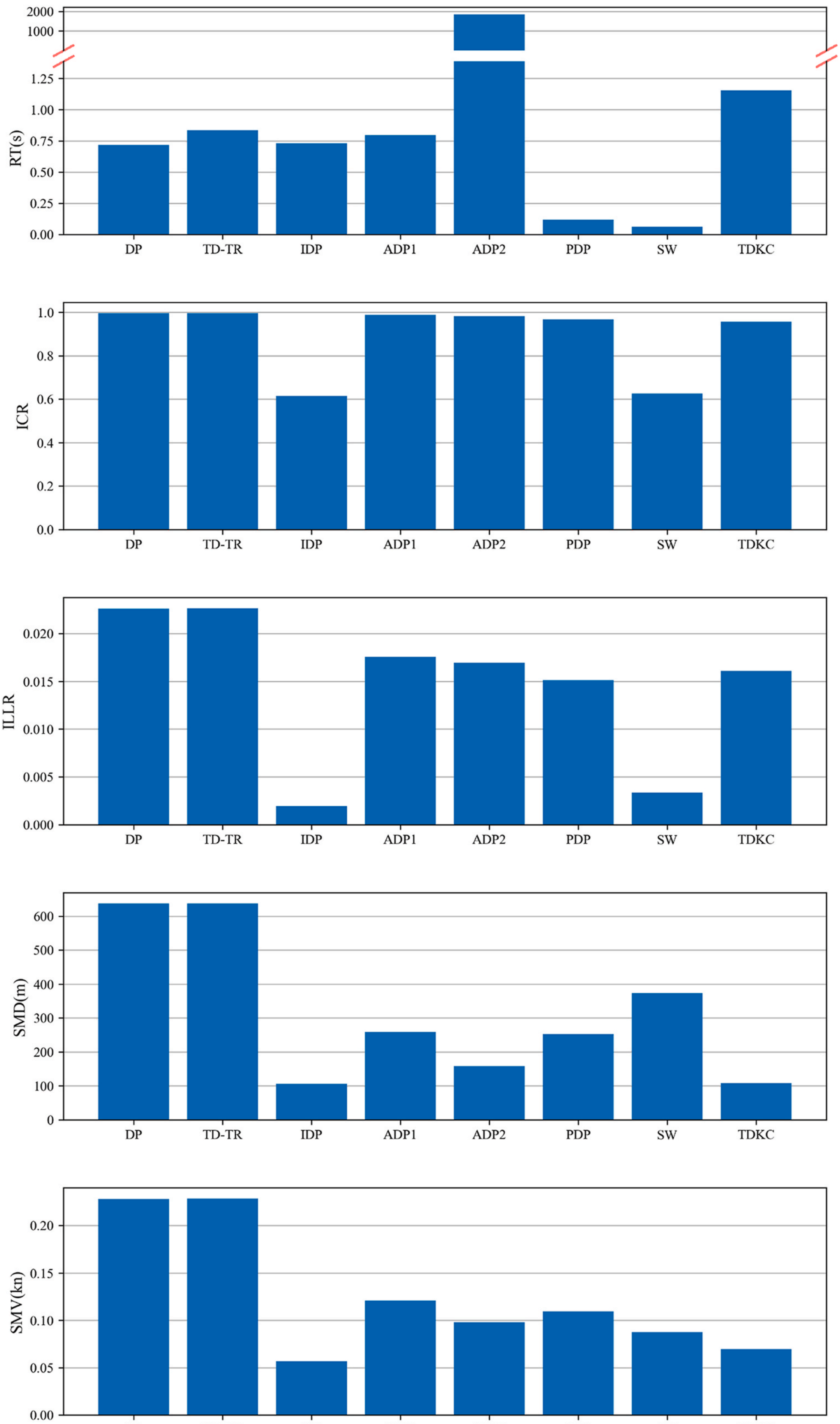


Fig. 13. Compression results of different algorithms.

time consumption around 0.750s. The proposed TDKC algorithm spends slightly more time, which is 1.156s. Besides, due to the training procedure on the entire dataset, ADP2 noticeably consumes a substantial amount of time.

Regarding the effectiveness of compression, all top-down algorithms except for IDP achieve similar ICRs spanning from 95.697% to 99.551%, whereas IDP and SW only compress 61.611% and 62.543% of the points. Within such a low compression rate, IDP and SW achieve the minimal ILLRs of 0.197% and 0.335% respectively, while others reach ILLRs ranging from 1.511% to 2.263%, so outperforming the other algorithms.

As for similarity metrics, DP and TD-TR exhibit notably greater values for SMD and SMV around 637.372m and 0.228kn, respectively. ADP1 and PDP show comparable performance as well by reducing these values to 255.846m and 0.115kn. ADP2 performs slightly better with values of 158.260m and 0.098kn. It is evident that IDP demonstrates the best performance on similarity metrics with the minimum SMD and SVD valuing at 106.086m and 0.057kn. Although SW algorithm has a similar ICR with IDP, it demonstrates unsatisfied performance on similarity metrics. On the contrary, with values of 107.905m and 0.070kn, TDKC algorithm with such a high compression rate exhibits rather close performance compared to IDP.

5.1.2. *Motion variation analysis*

To provide additional insight into the performance of feature preservation, Figs. 14–17 depict the compression performance in terms of position, speed variation over time, course variation over time, and velocity distribution. Evidence in Fig. 14 indicates that all algorithms proficiently retain the overall shape of the trajectory. However, DP, TD-

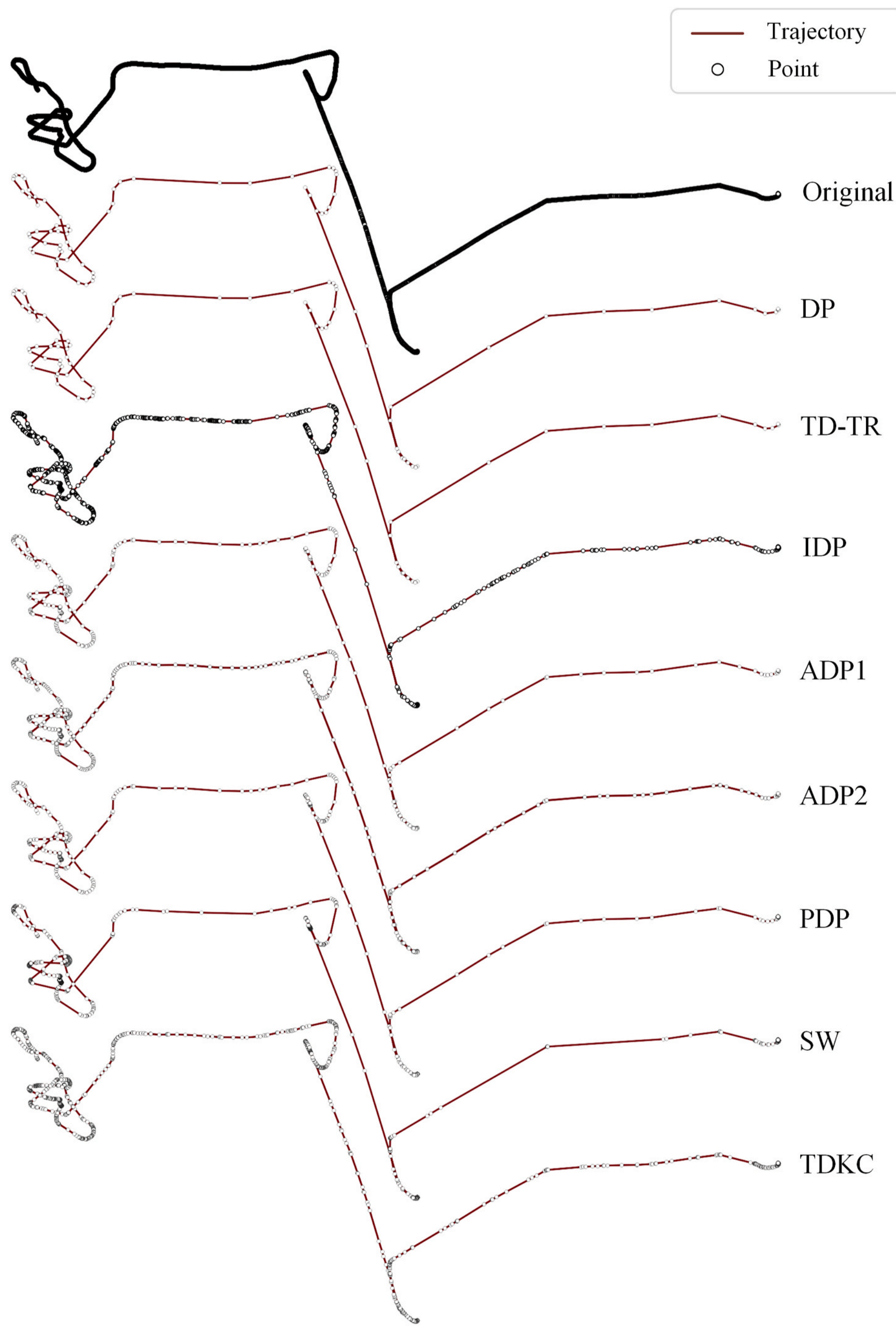


Fig. 14. Compressed trajectories of different algorithms.

TR, and SW identified fewer key points during the straight navigation parts compared to others. Notably, IDP, ADP2, and TDKC showcased a good capability in preserving a greater number of key points during these straight segments. In Fig. 15, it is observed that IDP and TDKC effectively maintained the variation features in speed, while others failed to preserve sudden changes of these features. Moreover, the course variations in Fig. 16 indicated that all algorithms retained the main changes in moving segments, while TDKC performs the most effectively in terms of capturing and retaining course variations in these segments.

However, in stationary segments the deviations were only well-captured by IDP and SW algorithms. PDP also identified a substantial number of key points in these segments. Conversely, other top-down algorithms only preserved a few points in stationary parts. When a ship is mooring, due to environmental forces, the course information will fluctuate frequently, which is a common limitation in AIS data processing. The distribution coverage portrayed in Fig. 17 shows that DP and TD-KC preserved only a few points in polar coordinates. ADP1, ADP2, and PDP preserved a wider range of points with speeds under 3 knots but struggled to capture trajectories with higher speeds. IDP, SW, and TDKC maintained the original distribution well, with TDKC achieving even better coverage while possessing a one-third higher compression rate than IDP and SW.

It is noteworthy that excessive retention of features related to ship stationary periods can impact the performance evaluation of compression algorithms. Although SW demonstrates strength in achieving a low ILLR, its high SMD and SMV values compared to TDKC imply their disregarding of global features. Additionally, IDP achieves the best performance on ILLR, SMD, and SMV. However, the retention of approximately one-third more points significantly widened the performance gap compared to TDKC. Therefore, we further checked the period when this ship was mooring at Paldiski South Harbor. As shown in Fig. 18, the reported position exhibits deviations within an extraordinarily small range, which is a common occurrence attributable to environmental forces typical in AIS report messages. Despite the ship being stationary, the phenomenon leads to a relatively lengthy trajectory. Both IDP and SW demonstrate a high degree of sensitivity to such scenarios, resulting in the retention of numerous superfluous points. PDP exhibits comparatively less sensitivity but still retains a considerable number of points. Other algorithms effectively compressed these fluctuated points to one or very few points. Though maintaining trajectory length under this circumstance contributes to a lower ILLR, its validity and necessity remain questionable since we know the ship is stationary at a certain place. Conclusively, TDKC successfully captures the key points of entering and exiting the stationary state.

In summary, the outperformance of TDKC demonstrates its ability to remove redundant data while maintaining key features among the whole trajectory. As kinematic information in trajectory data collectively defines the ship's path and behavior, analyzing preserved trajectory data after applying TDKC allows for the rapid and accurate detection of critical situations, such as near misses, that may signal potential risks, thereby leading to better insights into ship behaviors under risky scenarios, enhancing maritime operators' ability to manage and respond to potential risks effectively.

5.2. Results of various traffic areas

5.2.1. Compressed trajectories visualization

Fig. 19 compares the overall trajectories before and after applying compression algorithms. The comparison reveals minimal distinctions in the visualizations of the traffic flow obtained by these algorithms. Since the shape of visualized trajectories in the traffic flow only shows the results of positional information preserving, a comparison of all algorithms regarding their abilities for speed and course preservation is visualized in Figs. 20 and 21. Noticeably SOG and COG feature losses can be found in DP, TD-TR, ADP1, PDP, and SW algorithms. The traffic

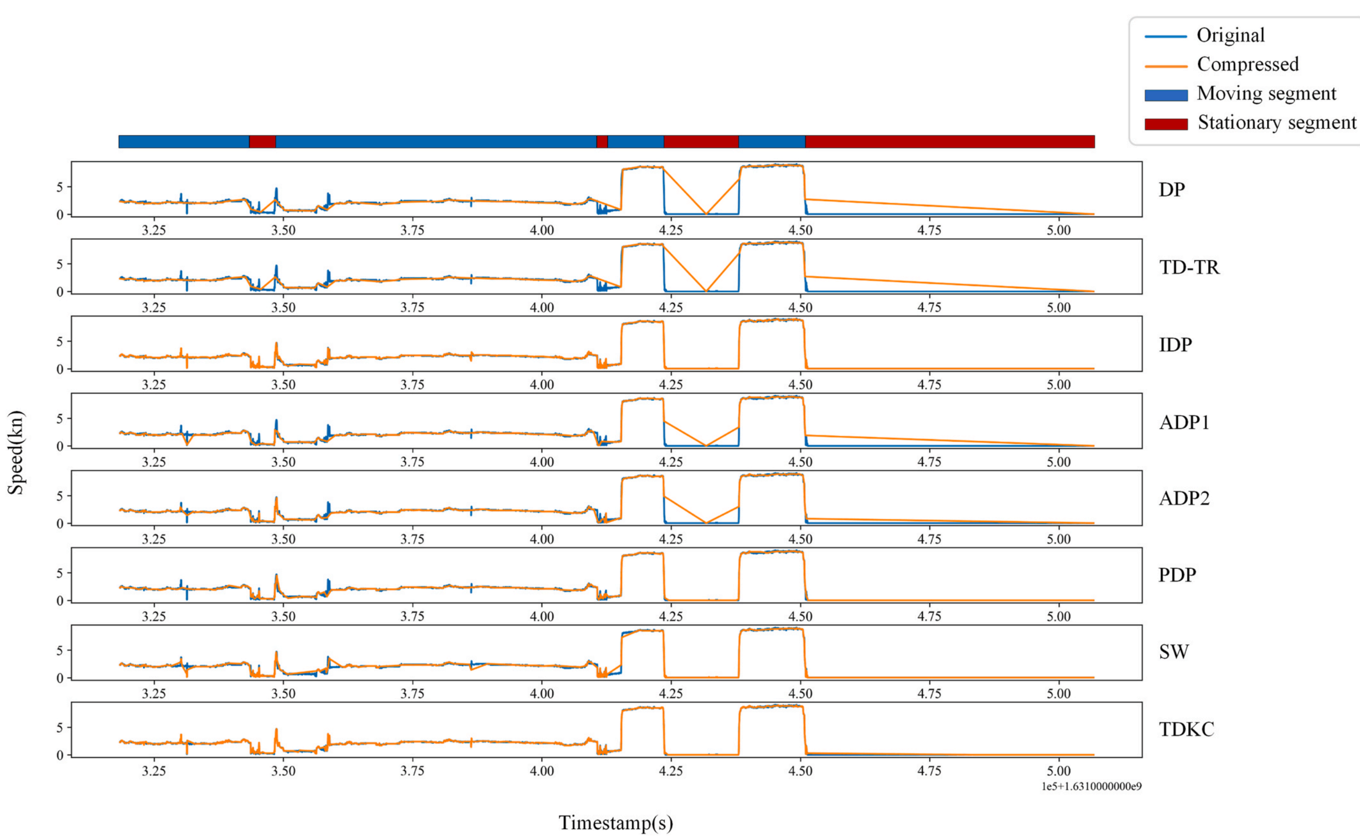


Fig. 15. Speed variation over time.

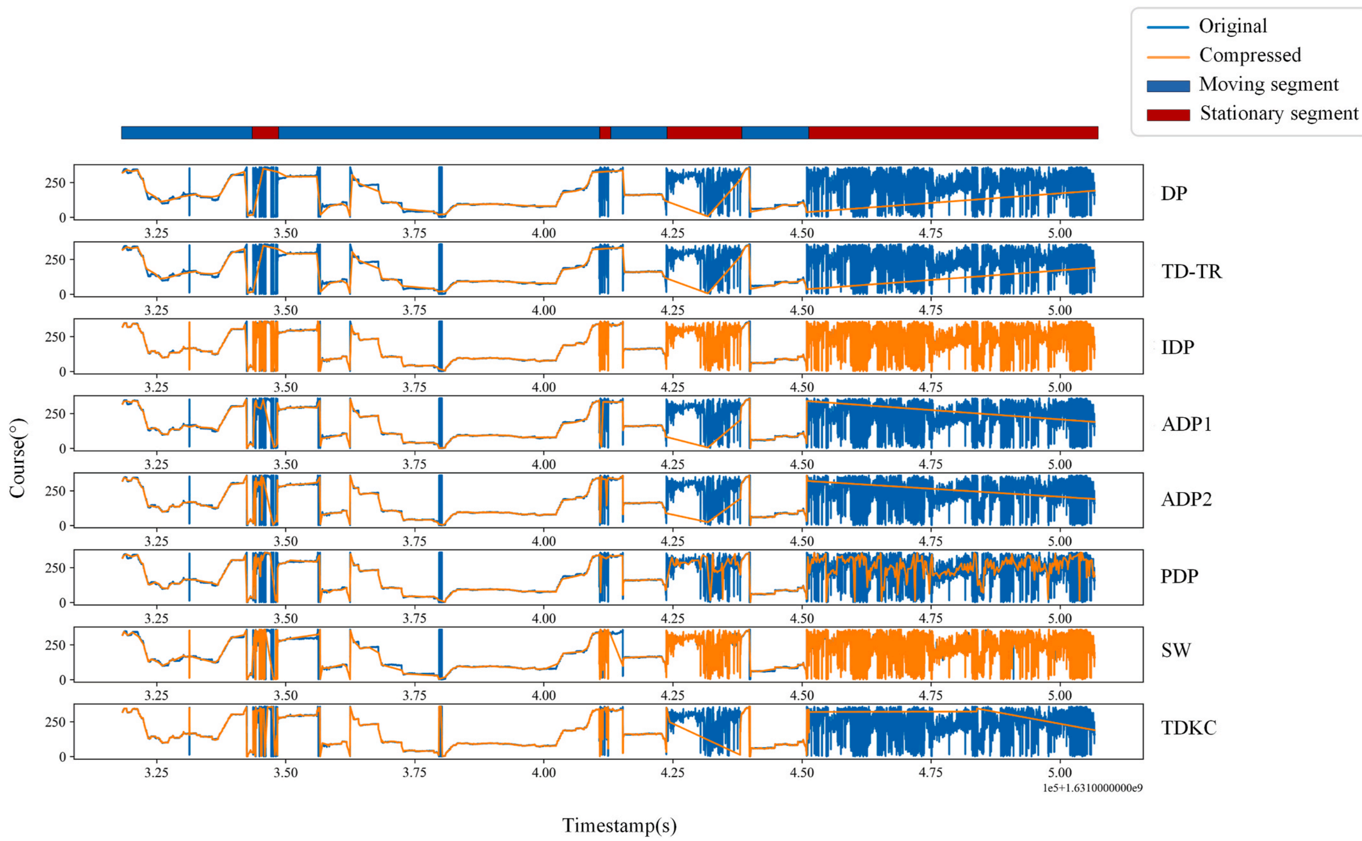


Fig. 16. Course variation over time.

patterns in the central and western regions of Gulf of Finland appear vague in the compression results produced by these algorithms. On the contrary, IDP, ADP2, and TDKC algorithm accurately retains both SOG and COG features. Yet, TDKC comprehensively incorporates time, position, speed, and course information, allowing accurate featuring of significant points of change in position, velocity, or both. The identification of such key features can benefit subsequent ship behavior and traffic analysis, such as elucidating more intricate interactions between the ships concerning the risk picture in the area using relevant analysis methods, thereby facilitating maritime practitioners to obtain better maritime situation awareness. To further demonstrate the superiority of our proposed algorithm, detailed discussions are elaborated in the subsequent sections based on evaluation metrics.

5.2.2. Regional analysis

Fig. 22 shows the execution time of each algorithm in separated areas. The overall trend depicted is consistent with the features presented in the results of individual trajectory. One can observe that SW demonstrates minimal RTs, spending 65.109s, 13.483s, 5.046s, 11.459s, 13.658s, and 21.295s respectively. PDP necessitates a slightly longer time, especially in overall area and Area 5 with durations of 179.635s and 98.739s. DP, TD-TR, IDP, and ADP1 show similar processing speeds except in Area 5, where IDP requires a significantly extended time at 389.884s. Apparently, ADP2 and TDKC appear the most time-consuming algorithms since they both exhaust all recursions inherently, whilst TDKC still possesses a shorter execution time than ADP2 apart from Area 5.

Fig. 23 displays the compression rate evaluation. The overview shows that IDP and SW continue to exhibit poor performance in efficiently compressing data. The trend observed in their curves suggests notably low OCRs in coastal and island areas, where the OCRs of SW even fall below 30%. On the other hand, all other top-down algorithms maintain OCRs above 90%. Among them, the consideration of comprehensive kinematic attributes renders TDKC to preserve slightly more points, with a value of 93.446% in overall area, and 95.136%, 95.183%, 93.294%, 93.635%, and 91.970% respectively from Area 1 to Area 5. The boxplots and violin plots of ICR demonstrate DP and TD-TR have rather concentrated distributions in all areas, which can be interpreted by their uniform and feature-agnostic threshold settings. ADP1, ADP2, PDP, and TDKC show slightly wider distributions, which are still relatively concentrated compared to IDP and SW. The distribution of IDP and SW implies their instability in compression efficiency.

In terms of LLR, Fig. 24 proves that SW performs best in keeping the length information of trajectories, demonstrating stabilized OLLR values ranging from 0.287% to 0.669% across different areas, which is significantly lower than other algorithms. IDP fails to exhibit superiority compared to SW in multi-trajectory analysis, but still shows improvement over other top-down algorithms, with LLR values from 0.413% to 5.245%. TDKC comes right behind IDP, the LLR of which varies from 0.540% to 8.427%.

Except for SW, the curves have experienced the same trend when the area is changed. The elevated OLLRs in Area 1, Area 2, and Area 3 validated our inference in the previous section, where we concluded that stationary segments will affect the performance of LLR. Furthermore, the distributions reveal that SW is more sensitive to stationary trajectories and segments compared to top-down algorithms, as also revealed during individual trajectory results analysis. IDP also exhibits relatively concentrated LLR without losing much length information. However, PDP no longer shows sensitivity in these areas owing to its top-down nature, which is even worse than ADP2 and TDKC.

The boxplots and violin plots of ILLR regarding top-down algorithms exhibit characteristics of bimodal distributions, especially in Area 2. Although most data have relatively low ILLR values, we can still see a considerable portion of the data distributed near 100% ILLR. Such a phenomenon can be caused by the presence of stationary trajectories. If the segment displayed in Fig. 18 forms a complete trajectory, it implies

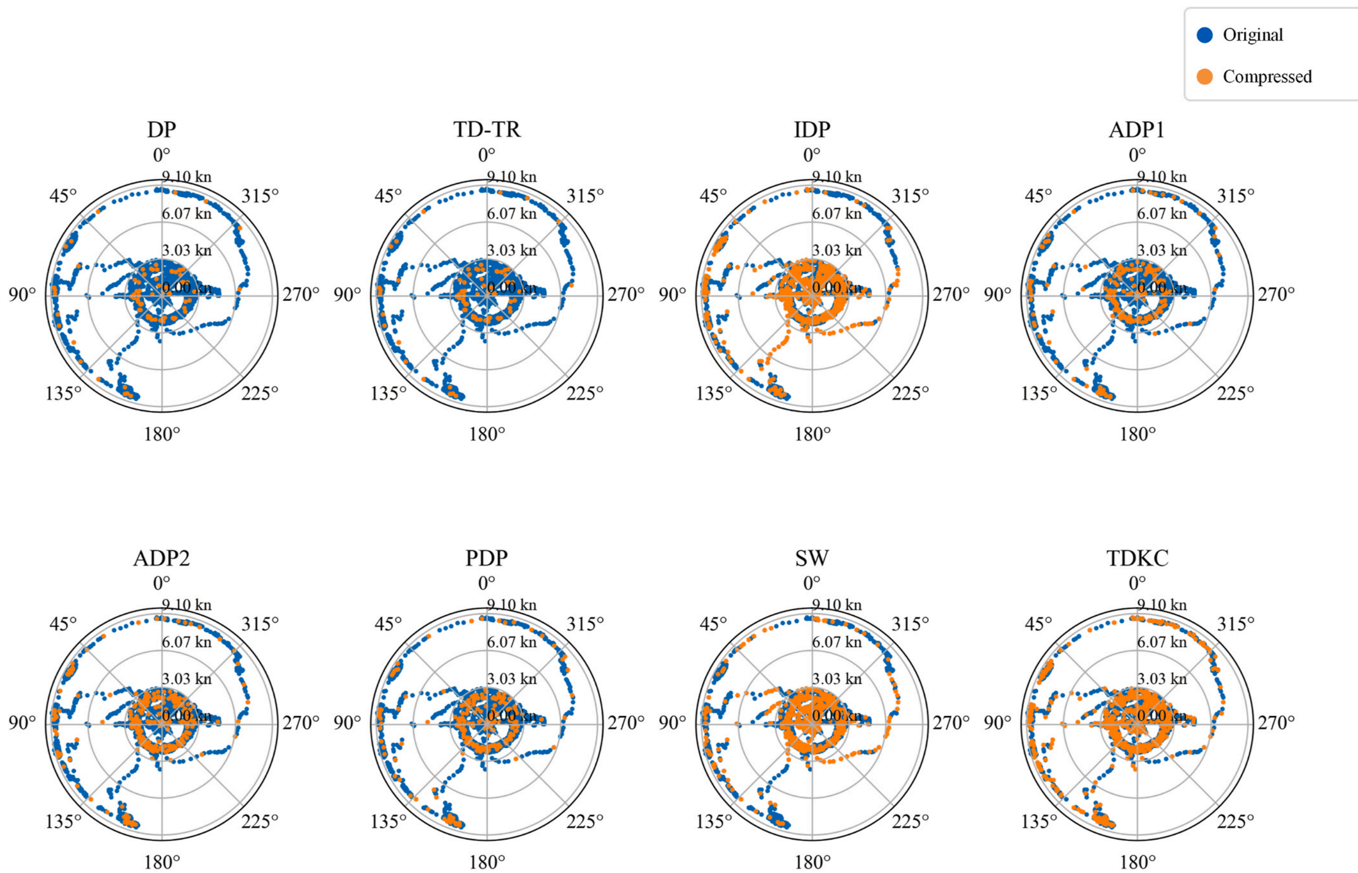


Fig. 17. Speed-course data preservation.

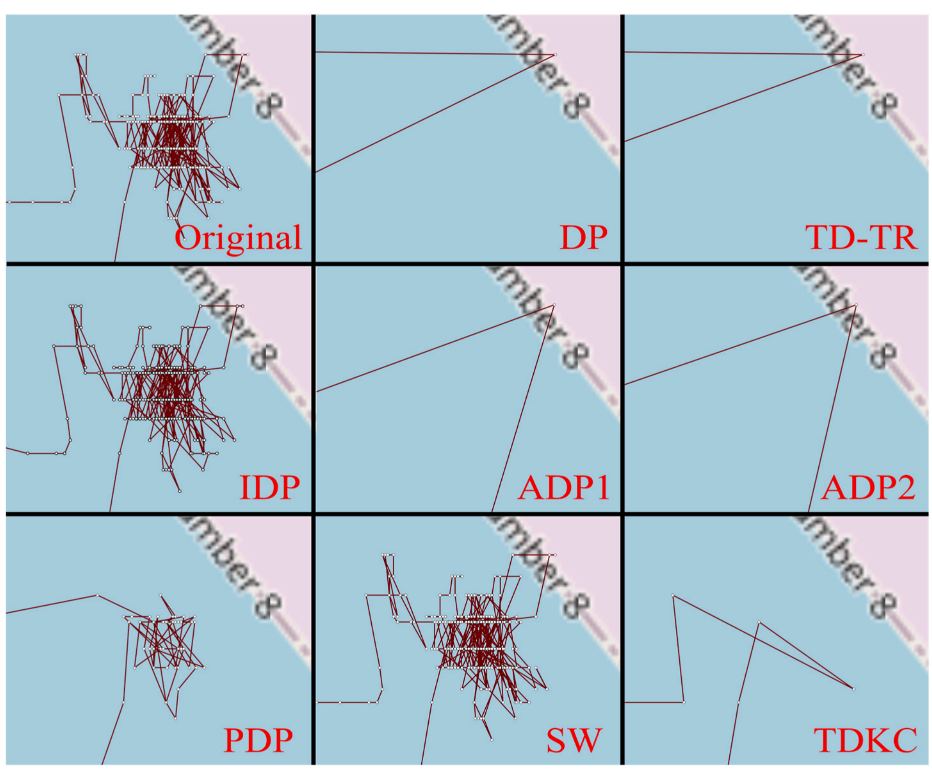


Fig. 18. Mooring segments.

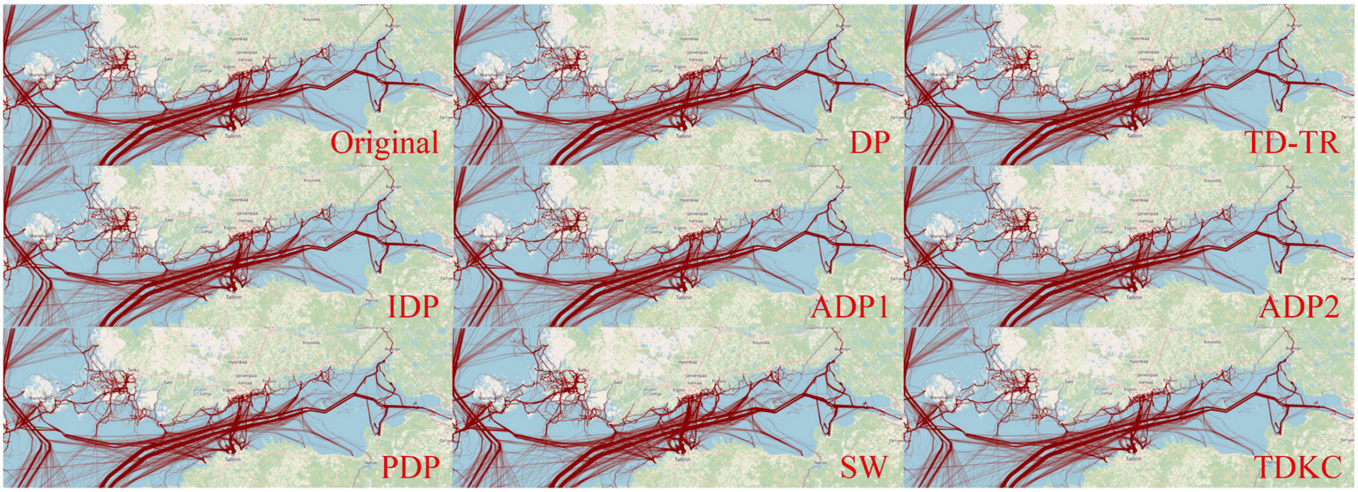


Fig. 19. Trajectory compression results in Gulf of Finland.

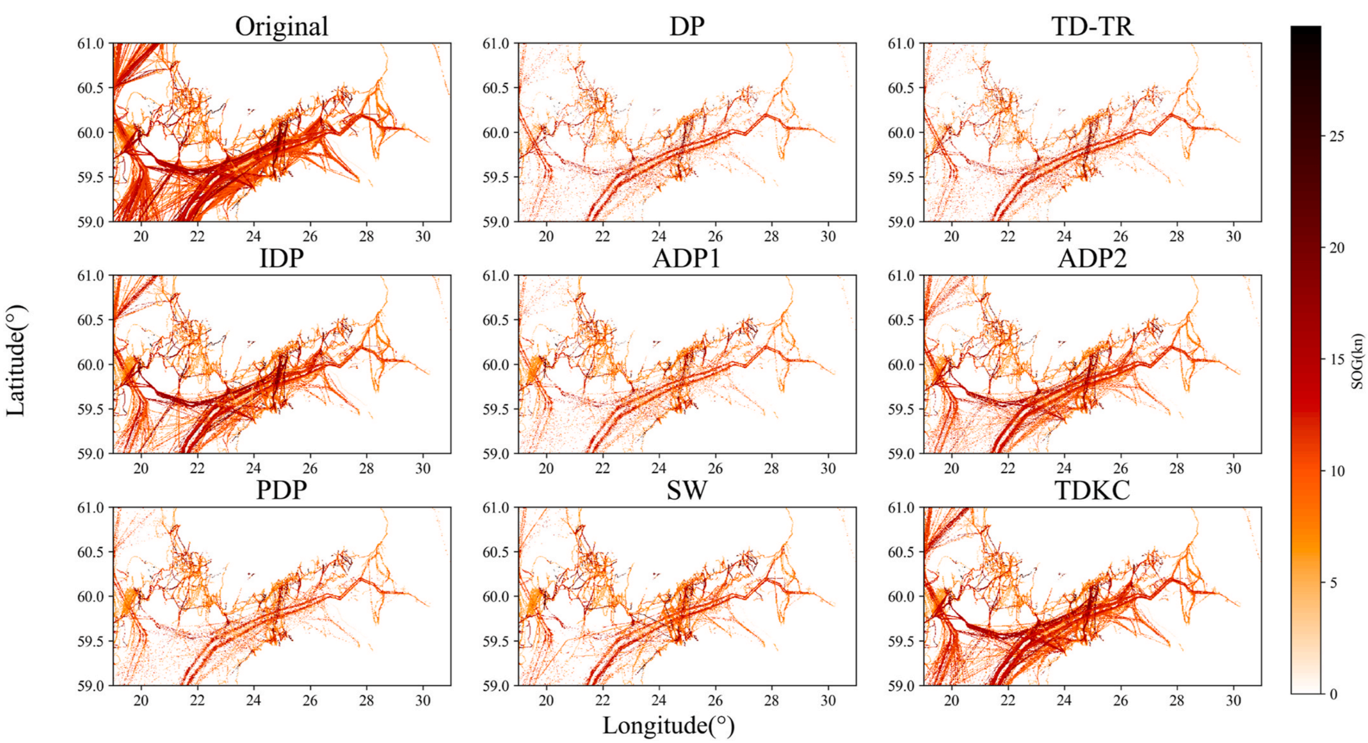


Fig. 20. Spatial SOG mapping.

that this trajectory does not contain any other moving segments. Fluctuations occurring in place might be compressed into a single point by a top-down algorithm. Consequently, the loss of length information for a single point compared to trajectory amounts to 100%. However, such a high rate does not necessarily mean poor performance since the stationary period is usually compressed and represented by few points (Sanchez-Heres, 2019). Besides, in Area 3, Area 4, and Area 5 almost all mean values of ILLR exceed the upper bound of the boxplots, which implies the influence of stationary trajectories and segments is non-negligible. Therefore, it is imperative to proceed with caution regarding the results of LLR assessment in areas that contain numerous stationary segments or trajectories.

Fig. 25 reveals that the proposed algorithm demonstrates significant advantages compared to others regarding SMD. The mean value overview curves show that DP, TD-TR, ADP1, PDP and SW demonstrate obviously higher SMD compared to IDP, ADP2, and TDKC. Although SW exhibits slight advancement compared to DP, TD-TR, ADP1, and PDP in Area 1, Area 2, and Area 3 due to its strength in identifying local features within relatively small ranges, the great deviations in Area 4 and Area 5 lead to its SMD reaching 2575.972m, suggesting poorer performance compared to top-down algorithms. In addition, the distribution plots also prove this deduction, showing SW's instability across different areas. Despite exhibiting rather focused distribution in Area 3, the maximum SMD of SW even reaches 19437.150m in Area 4. On the contrary, advanced algorithms such as IDP, ADP2, and TDKC demonstrate relatively consistent and dense distributions. Among them, TDKC shows a lower average SMD, performing notably well in Area 4 and Area 5 with mean SMDs of 340.175m and 298.015m and being comparable to the others in the rest areas. In addition, ADP1 and PDP demonstrate restricted enhancements compared to DP and TD-TR.

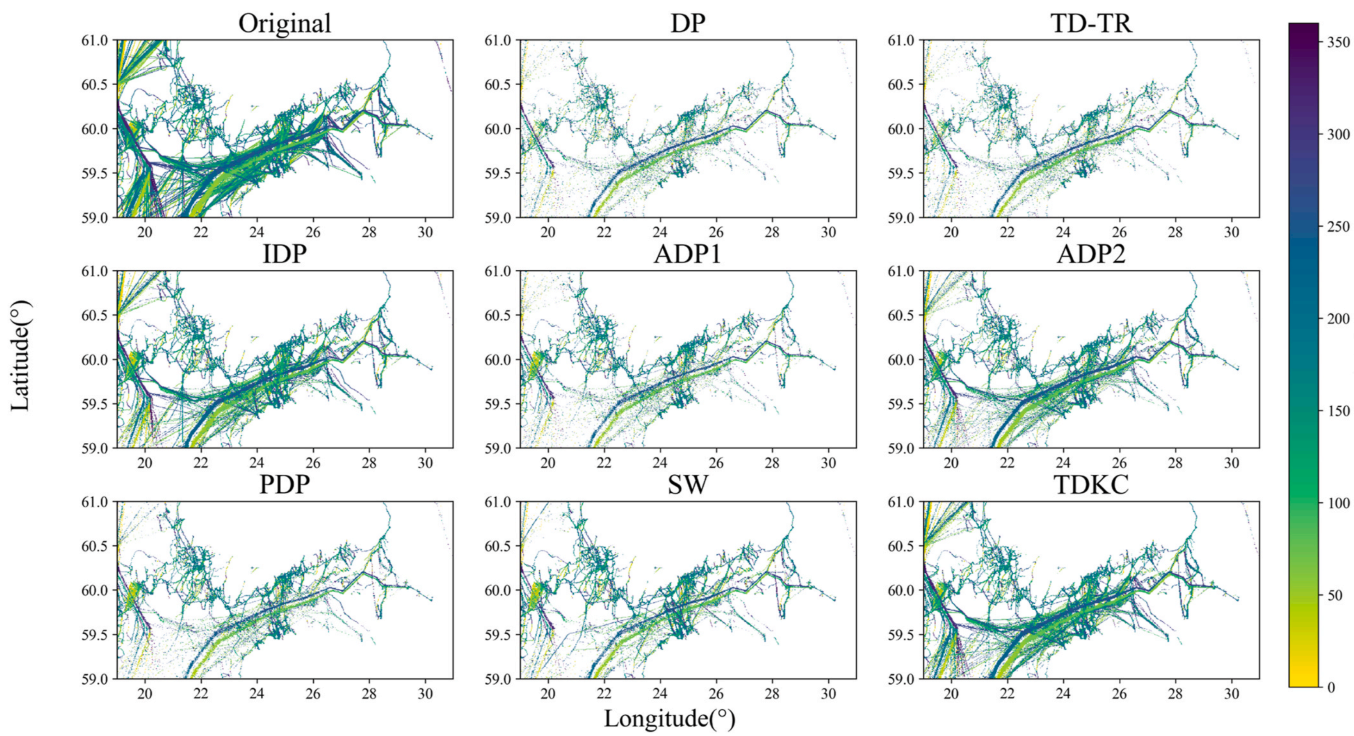


Fig. 21. Spatial COG mapping.

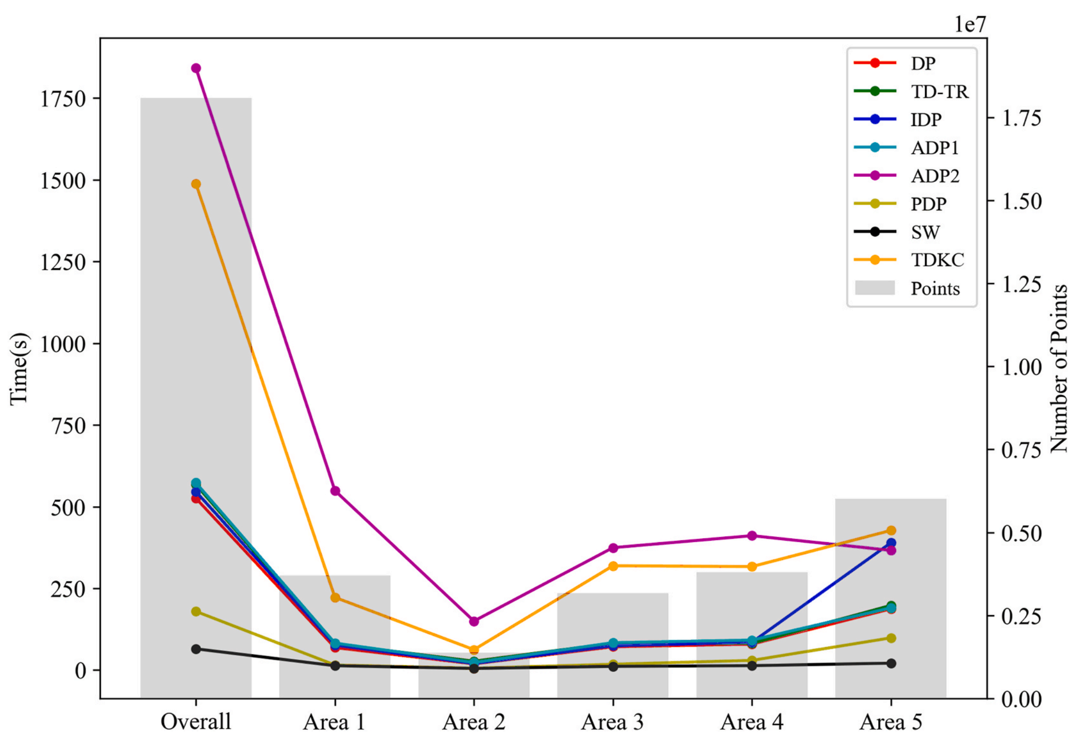


Fig. 22. Running time evaluation for algorithms on different areas.

Regarding SMV, the curves in Fig. 26 first illustrate that DP and TD-TR perform the poorest in speed and course information retention, with SMV values ranging from 0.610kn to 0.907kn. Subsequently, ADP1 and PDP show improvements by reducing the mean SMVs approximately to 0.457kn and 0.385kn respectively. For other algorithms, it becomes apparent that IDP exhibits the best performance with SMV spanning from 0.098kn to 0.178kn. ADP2, SW, and TDKC come after IDP, showing comparable results around 0.189kn. Nonetheless, TDKC demonstrates superior performance in Area 2, Area 4, and Area 5, exhibiting a slight inferiority to IDP. Additionally, the distributions of SMV indicate that all other algorithms achieve profound improvements compared to traditional DP and TD-TR algorithms. However, ADP1, PDP, and SW demonstrate limited advantages in open water areas. So, IDP, ADP2, and TDKC emerge as the most stable algorithms in preserving velocity

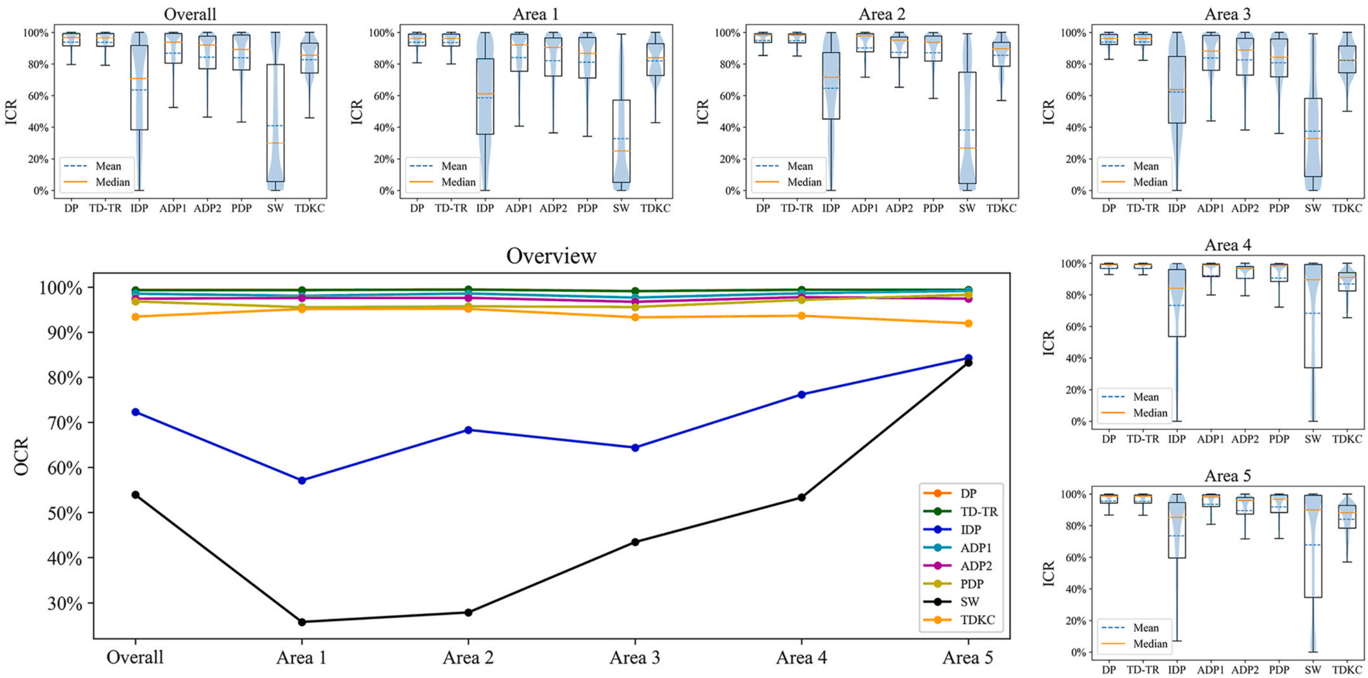


Fig. 23. Compression rate evaluation for algorithms in different areas.

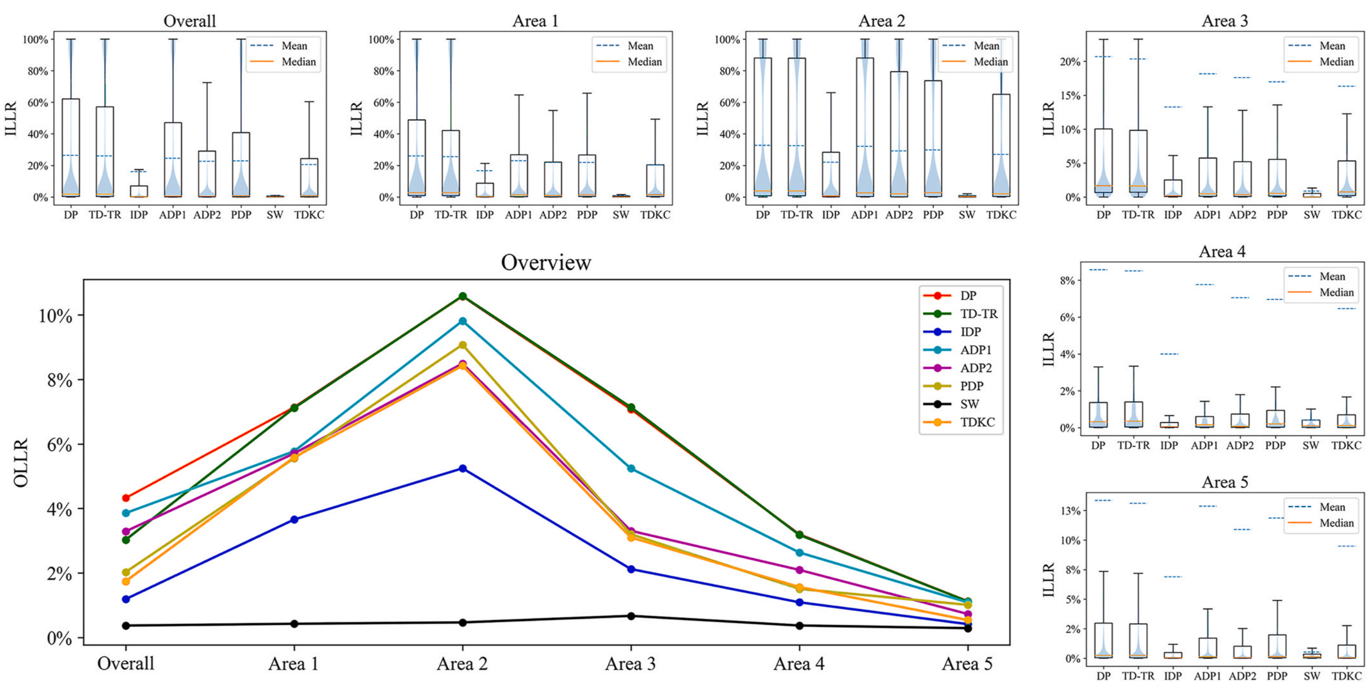


Fig. 24. LLR evaluation for algorithms on different areas.

information. Concluding, although TDKC shows a minor disadvantage due to its considerably higher compression rate compared to IDP, it slightly outperforms ADP2.

In conclusion, TDKC algorithm effectively maintains a rather low feature error while achieving a high compression efficiency both across the entire water area and within different types of sub-regions. In contrast, SW and IDP algorithms sacrifice substantial compression ability to achieve similar levels of feature preservation. Although DP, TD-TR, and ADP1 algorithms can compress a larger volume of trajectory data, they also lead to a corresponding loss of key features. As for ADP2, it shows a similar balanced performance as TDKC does, yet the high computation time renders it impractical for real-world applications, especially when the focus is on several individual trajectories, ADP2 always requires a training process on the whole dataset. Therefore, in maritime traffic analysis, the capability of TDKC can ensure an accurate representation of traffic patterns and ship behaviors with a reduced data volume, thus contributing to a more effective understanding of maritime traffic dynamics. Since TDKC can identify key points with different kinematic characteristics, further studies regarding scenario and ship behavior analysis using data processed by TDKC can enhance maritime situational awareness and safety management.

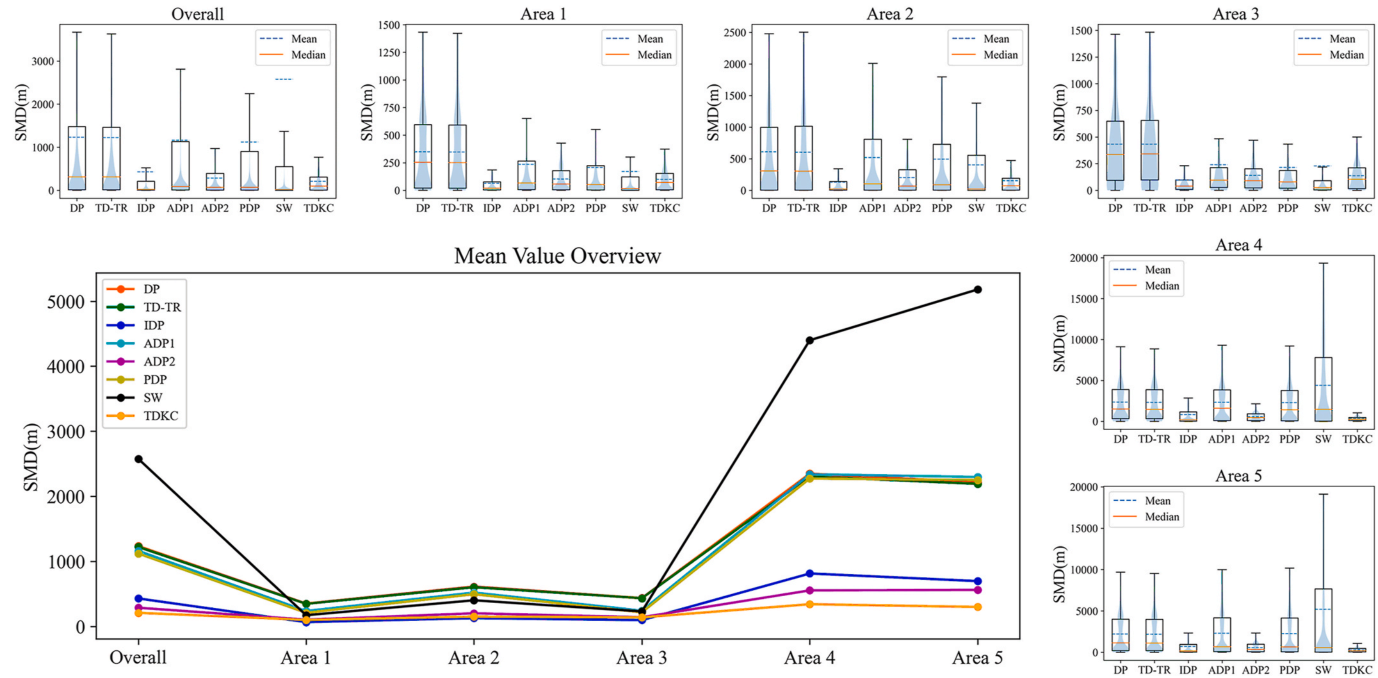


Fig. 25. SMD evaluation for algorithms in different areas.

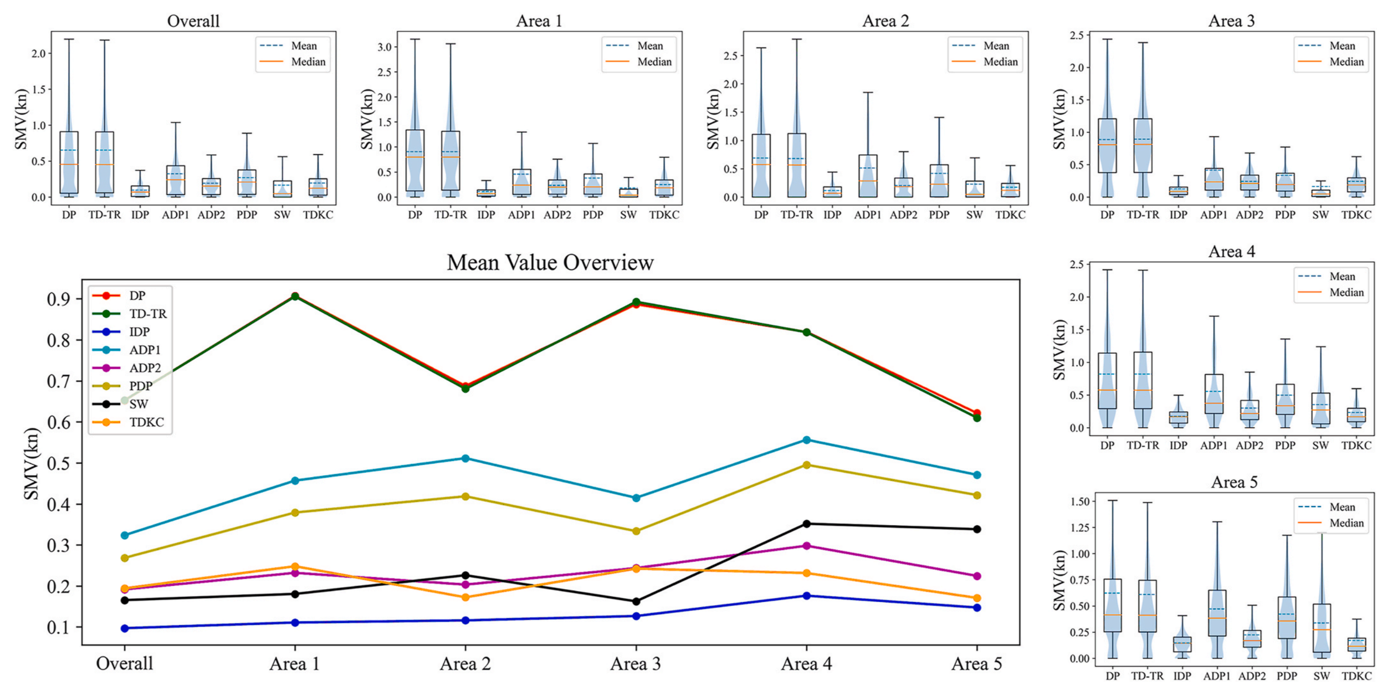


Fig. 26. SMV evaluation for algorithms in different areas.

5.2.3. *Trajectory sequence length analysis*

Trajectory sequence length appears to be another factor related to compression performances. As shown in Fig. 27, the RT curves demonstrate a growing trend when the sequence increases. TDKC is further proved to be time-consuming when the sequence length exceeds 70000. IDP and SW exhibit relatively stable curves and run the fastest. It should be noted that ADP2 is not considered in RT plot due to its necessity to be trained on the entire trajectory dataset before compression. As mentioned in Fig. 22, the training time is 1842.549s, much longer than any other algorithm. In addition, small valleys can be witnessed when the trajectory sequence length ranges from 50000 to 70000 for algorithms except for IDP and SW.

The ICR curves show that the majority of top-down algorithms manifest a rather higher compression rate than IDP and SW, which is consistent with results in the previous section. IDP and SW demonstrate downward patterns when the sequence length is lower than 60000. Prompted by the incomplete information represented by the limited number of trajectory points, it is noteworthy that all algorithms exhibit significant fluctuations at the beginning of the ICR curves. However, obvious valleys in IDP and SW curves, alongside minor peaks observed in other top-down algorithms emerge during the 50000-70000 interval.

Regarding ILLR, the abrupt curves for trajectories have 50000 to

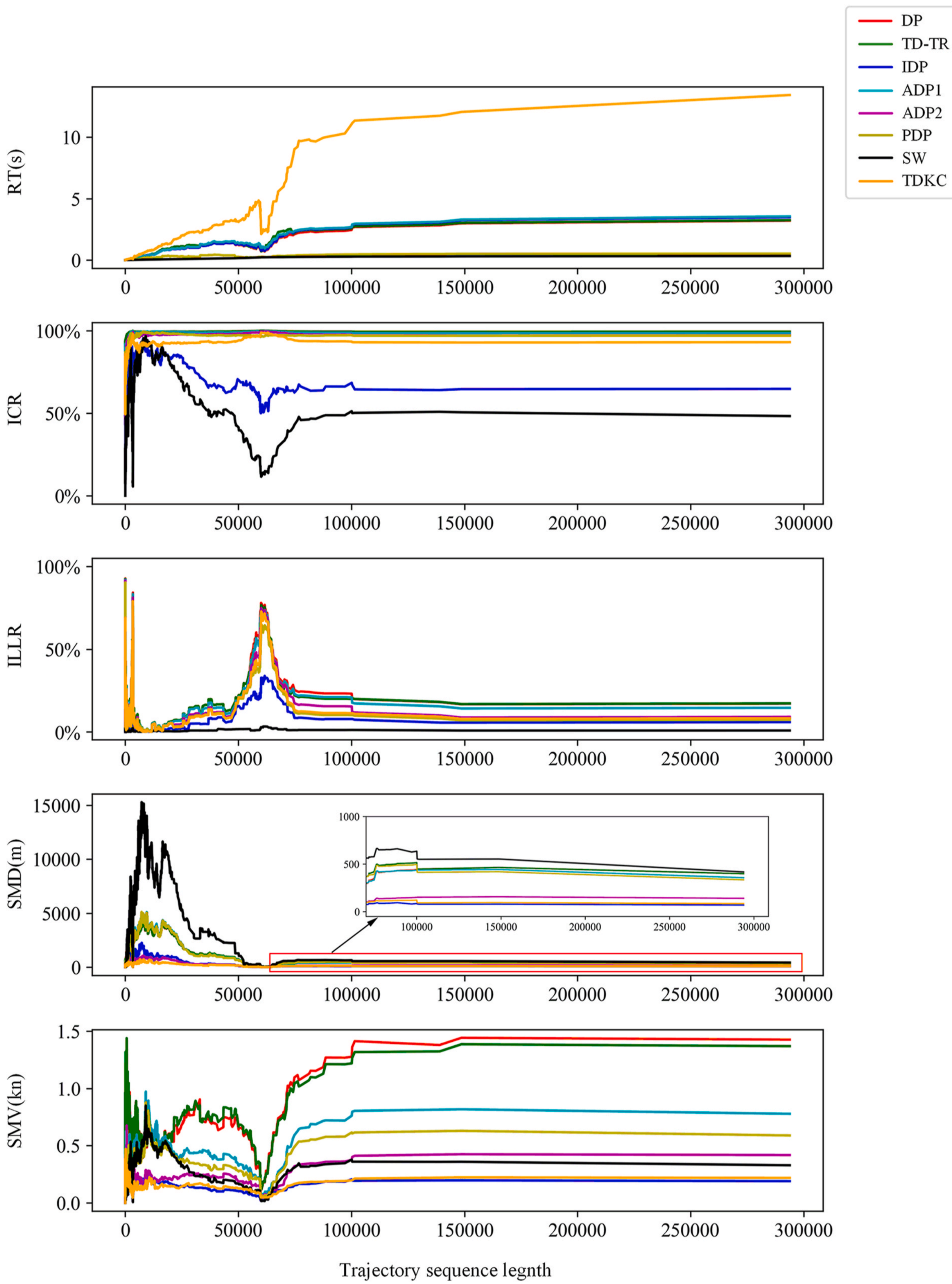


Fig. 27. Relationship between performance metrics and trajectory sequence length.

70000 points become more significant except for SW. Due to the fluctuated compression rate, another narrow sharp part emerges at the beginning of curves. Besides, SW exhibits a relatively low and consistent ILLR close to 1%. In contrast, others display a diminishing trend from the broad peak towards both ends.

The SMD plot displays that IDP, ADP2, and TDKC demonstrate superior performance than others. This distinction is particularly pronounced in the curve segment before 50000 trajectory points, where

TDKC performs best. A subtle transformation occurs thereafter, where the abrupt deviation parts in the aforementioned plots turn into small valleys and narrow the difference between curves. However, TDKC continues to exemplify its strength, ranking second with the curve stabilized at 83.9m, close to IDP settled at 69.1m. ADP2 ranks third and remains steady at 139.9m.

Finally, for SMV curves, it is evident that IDP and TDKC demonstrate significant superiority with most of the curves below 0.227kn. SW and ADP2, who rank third and fourth place, own SMV around 0.340 and 0.416 knots respectively. Additionally, the sharp deviation parts exhibit obvious valley features, further emphasizing the performance differences among different algorithms.

Fig. 28 and Table 8 shows the distribution analysis of trajectory sequence length, providing a reference to the RT quantification in our scoring system. They reveal that most trajectories in the water area contain fewer than 20000 points. For such a trajectory, the RT plot in Fig. 27 suggests that TDKC is still capable of processing most trajectories in a short time. Hence, we can infer that the execution time of TDKC in practical scenarios is tolerable. In addition, all plots in Fig. 27 exhibit fluctuations in the 50000-70000 interval. It is notable that the curves tend to show erratic patterns before the 50000-70000 interval but they stabilize thereafter. To explore the underlying reasons for the fluctuations, we further investigated the speed distribution of trajectories within 50000–70000 points. As shown in Fig. 29, most of the points possess a speed lower than 1 knot, indicating that these ships were swaying within a rather limited vicinity. Consequently, in top-down algorithms, points with extremely small deviations will be eliminated without being traversed, accounting for reductions in processing time as shown in the RT plot and slightly higher rates in the ICR plot, while for SW is the opposite. Moreover, such minimal deviations can also result in relatively minor SMD and SMV measurements, depicted by valleys in the SMD plot and SMV plot. On the other hand, reference to Fig. 18 clarifies the elimination of redundant points in a stationary state can cause a large LLR, which provides a clear explanation for peaks in the ILLR plot of Fig. 27.

Table 8
Trajectory with different sequence lengths.

Sequence length	Number of trajectories
0–20000	4103
50000-70000	84
Over 70000	98

5.3. *Algorithms comparison using scoring system*

To visually demonstrate the strengths and limitations of the algorithms, we conducted a quantitative evaluation of their performances using overall area results from Figs. 22–26 following the methodology of Section 3.4. Each algorithm's performance score is presented in Fig. 30. The main findings are discussed as follows.

DP and TD-TR present almost identical performance, while TD-TR only slightly outperforms on length information preservation. Although they perform well on CR, the universal threshold for DP and TD-TR causes termination during compression in advance, leading to the poorest performance on kinematic feature preservation. Despite TD-TR refining DP by considering time as a factor, it fails to deliver significant improvements for the algorithm.

IDP achieves significant improvements in feature preservation by incorporating a post-examination step that takes speed and course information into account. However, as a compression algorithm, such improvements might be counterproductive due to the considerable sacrifice in CR. Contrarily, the consideration of ship length enables ADP1 to adaptively compress trajectories. While maintaining a high CR,

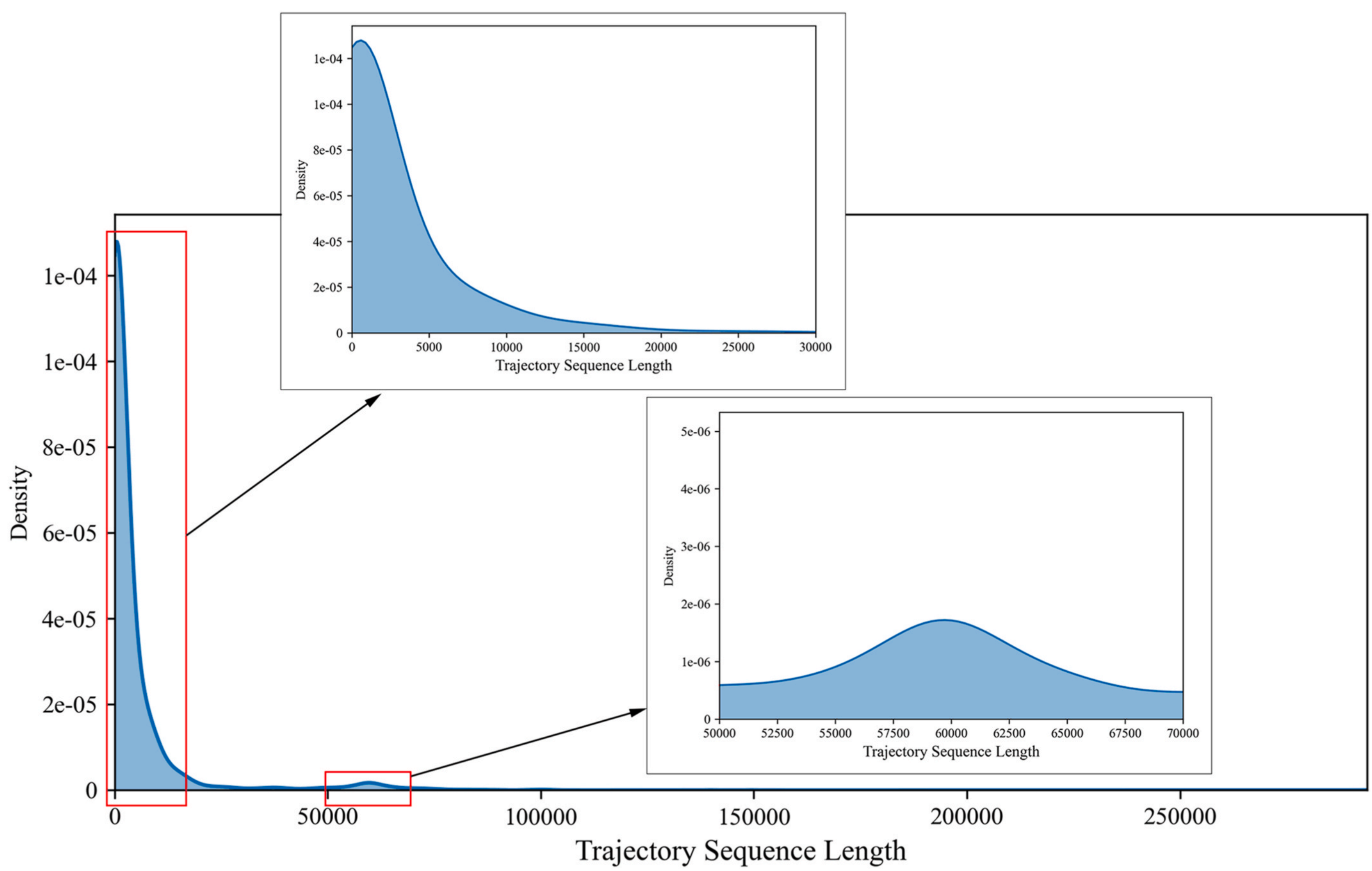


Fig. 28. Trajectory sequence length distribution.

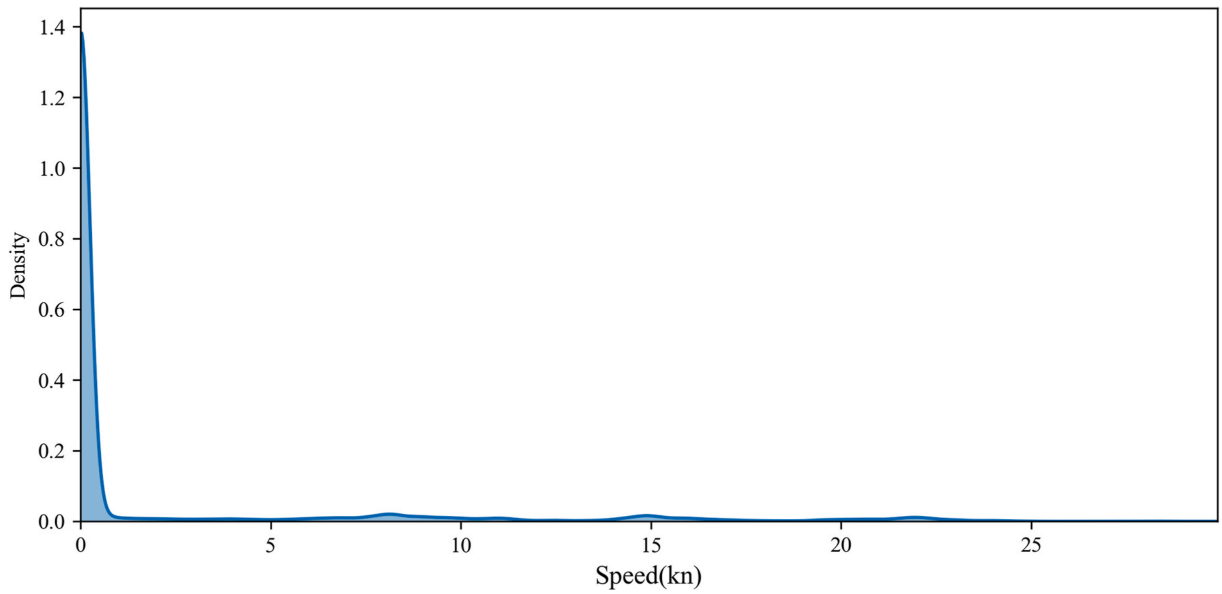


Fig. 29. Speed distribution of trajectories within 50000–70000 points.

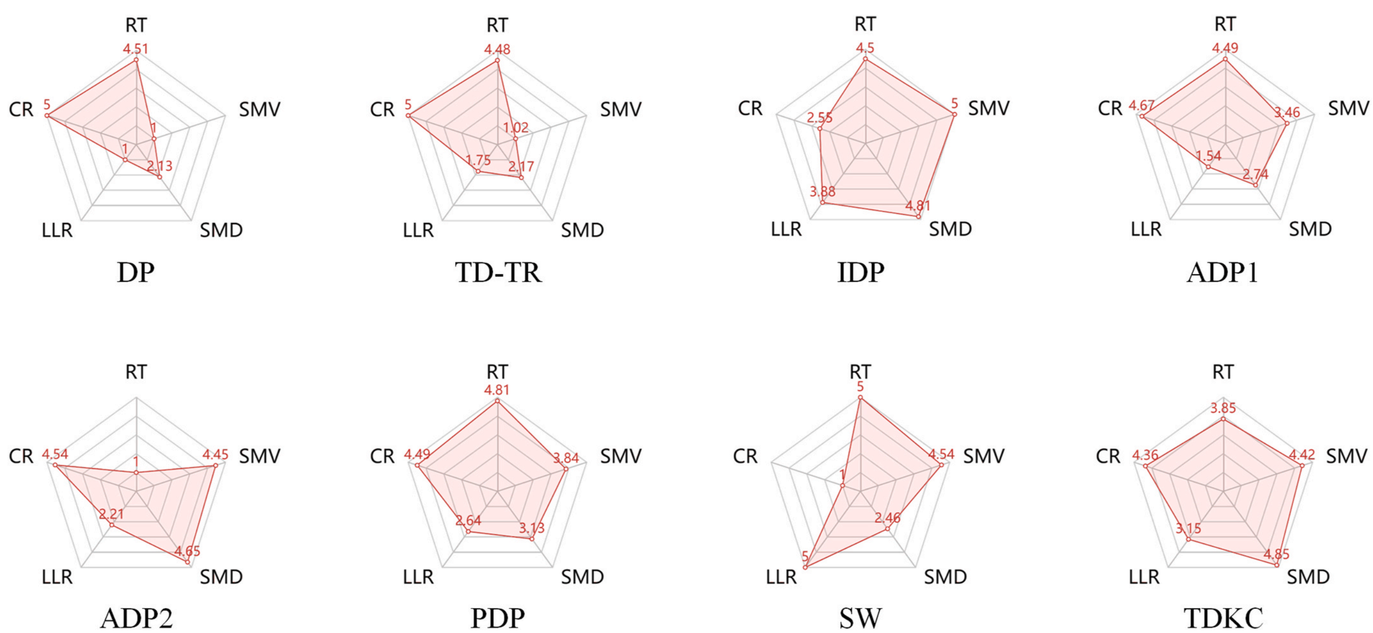


Fig. 30. Algorithm performance scores.

ADP1 also enhances the compression quality compared to DP and TD-TR algorithms. Nonetheless, the absence of velocity and ship length information limits such enhancement.

ADP2 endeavors to determine the optimal compression threshold by exhausting all recursions. Necessitating a training process in advance, ADP2 even requires more time and lacks applicability on individual trajectories. PDP demonstrates a more equitable performance, optimizing both the execution time and feature preservation ability of traditional top-down algorithms by pre-identifying key points. Yet, the inherent characteristics of being a top-down algorithm render PDP to demonstrate moderate performance on kinematic feature preservation. As for SW, although it shows strengths in length and velocity information preservation, its poor performance on CR and modest results on SMD suggest a careful assessment of SW's suitability for the compression task in certain situations.

Our proposed algorithm TDKC explores the optimal threshold without prematurely ending the recursive process, attempting to achieve high-quality results that retain as many trajectory features as possible with minimal points. Despite the increase in time overhead, such a trade-off enhances the accuracy and compression rate of TDKC algorithm, emerging as a superior compression algorithm attributed to its comprehensive consideration of time, position, speed, and course. Such superiority facilitates TDKC to obtain the most reliable results with commendable compression quality across diverse regions.

6. Conclusions

Research on trajectory compression techniques has facilitated the storage, processing, analyzing, transmitting, and transferring of maritime AIS data. In this study, a novel adaptive trajectory compression and feature preservation method for maritime traffic analysis was proposed. The performance of the proposed method was validated on 7-day AIS data from Gulf of Finland and compared to 7 other traditional and popular advanced methods. The dataset was divided according to various traffic areas to gain insights into the performance of each method. In addition, a novel metric for velocity-based similarity measurement was proposed. The main contributions of this study are summarized as follows:

The proposed method adopts SED and SVD to thoroughly leverage kinematic information within trajectories. The purpose of SVD is to compensate for the shortcomings of previous methods in considering speed and course. The construction of CBT effectively addresses the recursion termination problem. In combination with CBT, the adaptive threshold-setting strategy automatically selects the optimal thresholds based on the unique characteristics of each trajectory. This approach mitigates the inaccuracy associated with manual configuration, significantly enhancing the algorithm's versatility and enabling its flexible application across various water areas. By establishing a quantitative evaluation system, the experiment results indicate that the proposed method outperforms others in terms of effectively preserving key information while maintaining a high compression rate. In addition, the strengths and flaws of each method employed in the case study were also highlighted, providing insights into their respective application scenarios. Given its superiority over previous algorithms, TDKC is capable of reducing data volume, facilitating faster and more efficient transfer over networks. The preservation of essential trajectory features also allows faster analysis in trajectory analysis without a loss of accuracy. Therefore, our method is expected to significantly contribute to maritime traffic analysis.

The limitations of this study demonstrate the need to further enhance the computational performance of the proposed method. Despite the method incorporating a built-in running time optimization strategy and it can be powered by modern hardware devices to overcome this shortcoming, the algorithm structure is expected to be further optimized by omitting unnecessary recursion steps. Achieving the balance between its processing speed and commendable compression quality can be addressed in future steps. In addition, although we divided Gulf of Finland into five different water segments, more experimental analysis in other research areas should be performed to examine the applicability of the proposed method. Meanwhile, valuable research in relation to the identified trajectory features can be conducted in future works to facilitate the development of maritime industry.

CRediT authorship contribution statement

Shaoqing Guo: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Formal analysis, Data curation, Conceptualization. **Victor Bolbot:** Writing – review & editing, Supervision, Methodology, Conceptualization. **Osiris Valdez Banda:** Writing – review & editing, Supervision, Resources, Methodology, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research was financially supported by China Scholarship Council (Grant Number: 202206950019) and Enablers and Concepts for Automated Maritime Solutions (ECAMARIS) project which was partially funded by Business Finland. The authors want to thank the Baltic Marine Environment Protection Commission (Helsinki Commission, HELCOM) for the provision of AIS data related to the analyzed sea area.

Appendix

Table A1
List of abbreviations.

AIS	Automatic Identification System
ADP	Adaptive Douglas-Peucker
CBT	Compression Binary Tree
COG	Course over ground
CR	Compression rate
DOTS	Directed acyclic graph based Online Trajectory Simplification
DP	Douglas-Peucker
DTW	Dynamic Time Warping
GIS	Geographic Information System
GPS	Global Positioning System
ICR	Individual compression rate
IDP	Improved Douglas-Peucker
ILLR	Individual length loss rate
IQR	Interquartile Range
LLR	Length loss rate
MMSI	Maritime Mobile Service Identity
MRPA	Multiresolution Polygonal Approximation
OCR	Overall compression rate
OLLR	Overall length loss rate
PDP	Partition Douglas-Peucker
PED	Perpendicular Euclidean Distance
Q1	First quartile
Q3	Third quartile
RT	Running time
SED	Synchronous Euclidean Distance
SMD	Distance-based similarity
SMV	Velocity-based similarity
SOLAS	International Convention for the Safety of Life at Sea
SOG	Speed over ground

(*continued on next page*)

Table A1 (*continued*)

SQUISH	Spatial Quality Simplification Heuristic
STTrace	Spatiotemporal Trace
SVD	Synchronous Velocity Difference
SW	Sliding window
TDKC	Top-Down Kinematic Compression
TD-TR	Top-Down Time-Ratio

Table A2
List of variables and symbols.

c_i, c_i''	Course
$\Delta c_{i,j}$	Course change
d_i	Distance from the point to the baseline
d_{max}	Maximum distance from the point to the baseline
$d_{i,j}$	Distance between two points on the Earth's Surface
d_ε	Distance Threshold
end	End index of current processing trajectory's Points
i, j	Index
k	The number of discarded points
l_i	Left child
l, $\overline{p_i p_j}$	Baseline determined by p_i and p_j
lat_i	Latitude
lon_i	Longitude difference
m	The number of total trajectories
$\boldsymbol{m}_i$	Information difference vector
$\boldsymbol{m}_i^{norm}$	Normalized information difference vector
$\lvert \boldsymbol{m}_i^{norm} \rvert$	Magnitude of normalized information difference vector
$\lvert \boldsymbol{m}_{max}^{norm} \rvert$	Maximum magnitude of normalized information difference vector
$\max()$	The largest value in a given set
$\min()$	The smallest value in a given set
p_i	A ship trajectory point of T
p_i', p_i''	Projection points of p_i on the baseline
p_s	Start point
p_e	End point
p_{max}	Split point with maximum distance to the baseline
ped_i	PED
r_i	Right child
s_i, s_i''	Speed
sed_i	SED
sed_i^{norm}	Normalized SED
sed_ε	SED threshold
$start$	Start index of current processing trajectory's points
svd_i	SVD
svd_i^{norm}	Normalized SVD
svd_ε	SVD threshold
$score_{high}()$	Score function for positive metric
$score_{low}()$	Score function for negative metric
$\Delta s_{i,j}$	Speed change
t_i, t_i''	Timestamp
$\boldsymbol{v}_i$	Velocity vector
vx_i	Velocity component in x coordinate
vy_i	Velocity component in y coordinate
$\Delta \boldsymbol{v}_{i,j}$	Velocity change
w_j^{RT}	Weight for RT quantification
w_1^{CR}, w_2^{CR}, w_3^{CR}	Weight for CR quantification
w_1^{LLR}, w_2^{LLR}, w_3^{LLR}	Weight for LLR quantification
w_1^{SMD}, w_2^{SMD}, w_3^{SMD}	Weight for SMD quantification
w_1^{SMV}, w_2^{SMV}, w_3^{SMV}	Weight for SMV quantification
x_i, x_i', x_i''	x coordinate
y_i, y_i', y_i''	y coordinate
CBT	CBT of a trajectory
$D()$	Cumulative distance between two trajectory sequences
$ILLR$	ILLR of the trajectory
L, L', L_i, L_i'	Trajectory length
M	Metric value set
N_i	Node
OCR	OCR of the trajectory dataset
$OLLR$	OLLR of the trajectory dataset
QRT_i	Quantified RT
QCR_i	Quantified CR
$QLLR_i$	Quantified LLR
$QSMD_i$	Quantified SMD

(*continued on next page*)

Table A2 (*continued*)

$QSMV_i$	Quantified SMV
$Root$	Root node
S_{SED}	SED set of current processing trajectory
S_{SVD}	SVD set of current processing trajectory
$SubT_i$	A sub-trajectory of T
$SMD()$	SMD between trajectories
$SMV()$	SMV between trajectories
$SMD_{norm}()$	Normalized SMD between trajectories
$SMV_{norm}()$	Normalized SMV between trajectories
T	A ship trajectory
T'	Compressed ship trajectory of T
$V()$	Cumulative velocity difference between two trajectory sequences
α_i	Metric value
$\mu_{S_{SED}}$	Mean of S_{SED}
$\mu_{S_{SVD}}$	Mean of S_{SVD}
$\sigma_{S_{SED}}$	Standard deviations of S_{SED}
$\sigma_{S_{SVD}}$	Standard deviations of S_{SVD}
τ	The number of algorithms

References

Amigo, D., Pedroche, D.S., Garcia, J., Molina, J.M., 2021. Review and classification of trajectory summarisation algorithms: from compression to segmentation. Int. J. Distributed Sens. Netw. 17 https://doi.org/10.1177/15501477211050729.

Amigo, D., Pedroche, D.S., García, J., Molina, J.M., 2022. Segmentation optimization in trajectory-based ship classification. Journal of Computational Science 59, 101568. https://doi.org/10.1016/j.jocs.2022.101568.

Bai, X., Xie, Z., Xu, X., Xiao, Y., 2023. An adaptive threshold fast DBSCAN algorithm with preserved trajectory feature points for vessel trajectory clustering. Ocean Eng. 280, 114930 https://doi.org/10.1016/j.oceaneng.2023.114930.

Bellman, R., 1961. On the approximation of curves by line segments using dynamic programming. Commun. ACM 4, 284. https://doi.org/10.1145/366573.366611.

Bencs, L., Horemans, B., Buczyńska, A.J., Deutsch, F., Degraeuwe, B., Van Poppel, M., Van Grieken, R., 2020. Seasonality of ship emission related atmospheric pollution over coastal and open waters of the North Sea. Atmos. Environ. X 7, 100077. https://doi.org/10.1016/j.aeaoa.2020.100077.

Bolbot, V., Basnet, S., Zhao, H., Valdez Banda, O.A., Silverajan, B., 2022. Investigating a novel approach for cybersecurity risk analysis with application to remote pilotage operations. European Workshop on Maritime Systems Resilience and Security.

Cao, W., Li, Y., 2017. DOTS: an online and near-optimal trajectory simplification algorithm. J. Syst. Software 126, 34–44. https://doi.org/10.1016/j.jss.2017.01.003.

Chen, M., Xu, M., Franti, P., 2012. A fast O(N) multiresolution polygonal approximation algorithm for GPS trajectory simplification. IEEE Trans. Image Process. 21, 2770–2785. https://doi.org/10.1109/TIP.2012.2186146.

Chen, P., Van Gelder, P., Mou, J., 2019. Integration of elliptical ship domains and velocity obstacles for ship collision candidate detection. TransNav: International Journal on Marine Navigation and Safety of Sea Transportation 13, 751–758. https://doi.org/10.12716/1001.13.04.07.

De Vries, G.K.D., Van Someren, M., 2012. Machine learning for vessel trajectories using compression, alignments and domain knowledge. Expert Syst. Appl. 39, 13426–13439. https://doi.org/10.1016/j.eswa.2012.05.060.

Douglas, D.H., Peucker, T.K., 1973. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. Cartographica: Int. J. Geogr. Inf. Geovisualization 10, 112–122. https://doi.org/10.3138/FM57-6770-U75U-7727.

Du, L., Valdez Banda, O.A., Huang, Y., Goerlandt, F., Kujala, P., Zhang, W., 2021a. An empirical ship domain based on evasive maneuver and perceived collision risk. Reliab. Eng. Syst. Saf. 213, 107752 https://doi.org/10.1016/j.ress.2021.107752.

Du, Y., Zhang, X., Cao, Z., Wang, S., Liang, J., Zhang, F., Tang, J., 2021b. An optimized path planning method for coastal ships based on improved DDPG and DP. J. Adv. Transport. 2021 https://doi.org/10.1155/2021/7765130.

EMSA, 2019. EMSA 5-Year Strategy. European Maritime Safety Agency.

Gao, J., Cai, Z., Yu, W., Sun, W., 2023. Trajectory data compression algorithm based on ship navigation state and acceleration variation. J. Mar. Sci. Eng. 11 https://doi.org/10.3390/jmse11010216.

Gao, M., Shi, G.Y., 2019. Ship spatiotemporal key feature point online extraction based on AIS multi-sensor data using an improved sliding window algorithm. Sensors 19, 2706. https://doi.org/10.3390/s19122706.

Gu, Q., Zhen, R., Liu, J., Li, C., 2023. An improved RRT algorithm based on prior AIS information and DP compression for ship path planning. Ocean Eng. 279, 114595 https://doi.org/10.1016/j.oceaneng.2023.114595.

Guo, S., Bolbot, V., BahooToroody, A., Valdez Banda, O.A., Siow, C.L., 2023. Identification of hazardous encounter scenarios using AIS data for collision avoidance system testing. Advances in the Collision and Grounding of Ships and Offshore Structures. CRC Press.

Guo, S., Mou, J., Chen, L., Chen, P., 2021a. An anomaly detection method for AIS trajectory based on kinematic interpolation. J. Mar. Sci. Eng. 9 https://doi.org/10.3390/jmse9060609.

Guo, S., Mou, J., Chen, L., Chen, P., 2021b. Improved kinematic interpolation for AIS trajectory reconstruction. Ocean Eng. 234 https://doi.org/10.1016/j.oceaneng.2021.109256.

Han, Y., Sun, W., Zheng, B., 2017. COMPRESS: a comprehensive framework of trajectory compression in road networks. ACM Trans. Database Syst. 42, 1–49. https://doi.org/10.1145/3015457.

Huang, Y., Li, Y., Zhang, Z., Liu, R.W., 2020. GPU-accelerated compression and visualization of large-scale vessel trajectories in maritime IoT industries. IEEE Internet Things J. 7, 10794–10812. https://doi.org/10.1109/JIOT.2020.2989398.

Hwang, C.L., Yoon, K., 1981. Multiple Attribute Decision Making. Springer Berlin, Heidelberg.

IMO, 1974. International Convention of the Safety of Life at Sea (SOLAS). Author, London.

ITU, 2014. Recommendation ITU-R M. 1371-5. Technical Characteristics for an Automatic Identification System Using Time Division Multiple Access in the VHF Maritime Mobile Frequency Band.

Ji, Y., Qi, L., Balling, R., 2022. A dynamic adaptive grating algorithm for AIS-based ship trajectory compression. J. Navig. 75, 213–229. https://doi.org/10.1017/S0373463321000692.

Karney, C.F., 2013. Algorithms for geodesics. J. Geodesy 87, 43–55. https://doi.org/10.1007/s00190-012-0578-z.

Ke, B., Shao, J., Zhang, Y., Zhang, D., Yang, Y., 2016. An online approach for direction-based trajectory compression with error bound guarantee. In: Web Technologies and Applications: 18th Asia-Pacific Web Conference, APWeb 2016, Suzhou, China, September 23-25, 2016. Proceedings, Part I, pp. 79–91. Springer.

Keogh, E., Chu, S., Hart, D., Pazzani, M., 2004. Segmenting time series: a survey and novel approach. Data Mining in Time Series Databases. World Scientific.

Lee, J.S., Lee, H.T., Cho, I.S., 2022. Maritime traffic route detection framework based on statistical density analysis from AIS data using a clustering algorithm. IEEE Access 10, 23355–23366. https://doi.org/10.1109/ACCESS.2022.3154363.

Lee, W., Cho, S.W., 2022. AIS trajectories simplification algorithm considering topographic information. Sensors 22. https://doi.org/10.3390/s22187036.

Leichsenring, Y.E., Baldo, F., 2020. An evaluation of compression algorithms applied to moving object trajectories. Int. J. Geogr. Inf. Sci. 34, 539–558. https://doi.org/10.1080/13658816.2019.1676430.

Leodolter, M., Plant, C., Brändle, N., 2021. IncDTW: an R package for incremental calculation of dynamic time warping. J. Stat. Software 99, 1–23. https://doi.org/10.18637/jss.v099.i09.

Li, H., 2021. Typical trajectory extraction method for ships based on ais data and trajectory clustering. In: 2021 2nd International Conference on Artificial Intelligence and Information Systems, pp. 1–8.

Li, H., Lam, J.S.L., Yang, Z., Liu, J., Liu, R.W., Liang, M., Li, Y., 2022. Unsupervised hierarchical methodology of maritime traffic pattern extraction for knowledge discovery. Transport. Res. C Emerg. Technol. 143 https://doi.org/10.1016/j.trc.2022.103856.

Li, Y., Liu, R.W., Liu, J., Huang, Y., Hu, B., Wang, K., 2016. Trajectory compression-guided visualization of spatio-temporal AIS vessel density. In: 2016 8th International Conference on Wireless Communications & Signal Processing (WCSP).

Li, Y., Ren, H., 2022. Visual analysis of vessel behaviour based on trajectory data: a case study of the Yangtze River Estuary. ISPRS Int. J. Geo-Inf. 11 https://doi.org/10.3390/ijgi11040244.

Liu, C., Zhang, S., Cao, L., Lin, B., 2023. The identification of ship trajectories using multi-attribute compression and similarity metrics. J. Mar. Sci. Eng. 11, 2005. https://doi.org/10.3390/jmse11102005.

Liu, J., Li, H., Yang, Z., Wu, K., Liu, Y., Liu, R.W., 2019. Adaptive douglas-peucker algorithm with automatic thresholding for AIS-based vessel trajectory compression. IEEE Access 7, 150677–150692. https://doi.org/10.1109/ACCESS.2019.2947111.

Liu, S., Chen, G., Wei, L., Li, G., 2021. A novel compression approach for truck GPS trajectory data. IET Intell. Transp. Syst. 15, 74–83. https://doi.org/10.1049/itr2.12005.

Liu, Z., Yuan, W., Liang, M., Zhang, M., Liu, C., Liu, R.W., Liu, J., 2024. An online method for ship trajectory compression using AIS data. J. Navig. 1–22. https://doi.org/10.1017/S0373463324000171.

Ma, H., Zuo, Y., Li, T., 2022. Vessel navigation behavior analysis and multiple-trajectory prediction model based on AIS data. J. Adv. Transport. 2022 https://doi.org/10.1155/2022/6622862.

Makris, A., da Silva, C.L., Bogorny, V., Alvares, L.O., Macedo, J.A., Tserpes, K., 2021a. Evaluating the effect of compressing algorithms for trajectory similarity and classification problems. GeoInformatica 25, 679–711. https://doi.org/10.1007/s10707-021-00434-1.

Makris, A., Kontopoulos, I., Alimisis, P., Tserpes, K., 2021b. A comparison of trajectory compression algorithms over AIS data. IEEE Access 9, 92516–92530. https://doi.org/10.1109/ACCESS.2021.3092948.

Meratnia, N., de By, R.A., 2003. A new perspective on trajectory compression techniques. Proc. ISPRS Commission II and IV, WG II/5, II/6, IV/1 and IV/2 Joint Workshop Spatial, Temporal and Multi-Dimensional Data Modelling and Analysis.

Meratnia, N., de By, R.A., 2004. Spatiotemporal compression techniques for moving point objects. In: Advances in Database Technology-EDBT 2004: 9th International Conference on Extending Database Technology, Heraklion, Crete, Greece, March 14-18, 2004 9, pp. 765–782. Springer.

Muckell, J., Hwang, J.H., Patil, V., Lawson, C.T., Ping, F., Ravi, S.S., 2011. SQUISH: an online approach for GPS trajectory compression. 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., Ravi, S.S., 2014. Compression of trajectory data: a comprehensive evaluation and new approach. GeoInformatica 18, 435–460. https://doi.org/10.1007/s10707-013-0184-0.

Potamias, M., Patroumpas, K., Sellis, T., 2006. Sampling trajectory streams with spatiotemporal criteria. In: 18th International Conference on Scientific and Statistical Database Management (SSDBM'06). IEEE, pp. 275–284.

Qian, H., Lu, Y., 2017. Simplifying GPS trajectory data with enhanced spatial-temporal constraints. ISPRS Int. J. Geo-Inf. 6 https://doi.org/10.3390/ijgi6110329.

Rong, H., Teixeira, Â.P., Guedes Soares, C., 2020. Data mining approach to shipping route characterization and anomaly detection based on AIS data. Ocean Eng. 198 https://doi.org/10.1016/j.oceaneng.2020.106936.

Sanchez-Heres, L.F., 2019. Simplification and event identification for AIS trajectories: the equivalent passage plan method. J. Navig. 72, 307–320. https://doi.org/10.1017/S037346331800067X.

Sandu Popa, I., Zeitouni, K., Oria, V., Kharrat, A., 2015. Spatio-temporal compression of trajectories in road networks. GeoInformatica 19, 117–145. https://doi.org/10.1007/s10707-014-0208-4.

Shi, J., Liu, Z., 2022. Track pairs collision detection with applications to ship collision risk assessment. J. Mar. Sci. Eng. 10 https://doi.org/10.3390/jmse10020216.

Singh, A.K., Aggarwal, V., Saxena, P., Prakash, O., 2017. Performance analysis of trajectory compression algorithms on marine surveillance data. In: 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI).

Sinnott, R.W., 1984. Virtues of the haversine. Sky Telescope 68, 158.

Sousa, R.S.D., Boukerche, A., Loureiro, A.A.F., 2021. Vehicle trajectory similarity. ACM Comput. Surv. 53, 1–32. https://doi.org/10.1145/3406096.

Sun, P., Xia, S., Yuan, G., Li, D., 2016. An overview of moving object trajectory compression algorithms. Math. Probl Eng. 2016 https://doi.org/10.1155/2016/6587309.

Sun, S., Chen, Y., Piao, Z., Zhang, J., 2020. Vessel AIS trajectory online compression based on scan-pick-move algorithm added sliding window. IEEE Access 8, 109350–109359. https://doi.org/10.1109/ACCESS.2020.3001934.

Svanberg, M., Santén, V., Hörteborn, A., Holm, H., Finnsgård, C., 2019. AIS in maritime research. Mar. Pol. 106 https://doi.org/10.1016/j.marpol.2019.103520.

Tang, C., Wang, H., Zhao, J., Tang, Y., Yan, H., Xiao, Y., 2021. A method for compressing AIS trajectory data based on the adaptive-threshold Douglas-Peucker algorithm. Ocean Eng. 232 https://doi.org/10.1016/j.oceaneng.2021.109041.

Tang, J., Liu, L., Wu, J., 2019. A trajectory partition method based on combined movement features. Wireless Commun. Mobile Comput. 2019 https://doi.org/10.1155/2019/7803293.

Tavakoli, S., Khojasteh, D., Haghani, M., Hirdaris, S., 2023. A review on the progress and research directions of ocean engineering. Ocean Eng. 272, 113617 https://doi.org/10.1016/j.oceaneng.2023.113617.

Toohey, K., Duckham, M., 2015. Trajectory similarity measures. Sigspatial Special 7, 43–50. https://doi.org/10.1145/2782759.2782767.

Trajcevski, G., Cao, H., Scheuermanny, P., Wolfsonz, O., Vaccaro, D., 2006. On-line data reduction and the quality of history in moving objects databases. Proceedings of the 5th ACM International Workshop on Data Engineering for Wireless and Mobile Access, pp. 19–26.

Wang, T., 2013. An online data compression algorithm for trajectories (An OLDCAT). International Journal of Information and Education Technology 3, 480. https://doi.org/10.7763/IJIET.2013.V3.322.

Wei, Z., Xie, X., Zhang, X., 2020. AIS trajectory simplification algorithm considering ship behaviours. Ocean Eng. 216, 108086 https://doi.org/10.1016/j.oceaneng.2020.108086.

Wu, Y., Pelot, R., 2007. Comparison of simplifying line algorithms for recreational boating trajectory dedensification. Geomatics Solutions for Disaster Management.

Xin, X., Liu, K., Yang, Z., Zhang, J., Wu, X., 2021. A probabilistic risk approach for the collision detection of multi-ships under spatiotemporal movement uncertainty. Reliab. Eng. Syst. Saf. 215, 107772 https://doi.org/10.1016/j.ress.2021.107772.

Yan, R., Mo, H., Yang, D., Wang, S., 2022. Development of denoising and compression algorithms for AIS-based vessel trajectories. Ocean Eng. 252 https://doi.org/10.1016/j.oceaneng.2022.111207.

Yan, X., He, J., Ren, Q., Bai, C., Zhang, C., Wang, C., 2022. Research on extraction method of multiple narrow channel vessel trajectory feature in Yangtze River based on VITS data. J. Adv. Transport. 2022 https://doi.org/10.1155/2022/6533223.

Yang, D., Wu, L., Wang, S., Jia, H., Li, K.X., 2019. How big data enriches maritime research–a critical review of automatic identification system (AIS) data applications. Transport Rev. 39, 755–773. https://doi.org/10.1080/01441647.2019.1649315.

Yin, Z., Yang, D., Bai, X., 2022. Vessel destination prediction: a stacking approach. Transport. Res. C Emerg. Technol. 145, 16. https://doi.org/10.1016/j.trc.2022.103951.

Ziv, J., Lempel, A., 1978. Compression of individual sequences via variable-rate coding. IEEE Trans. Inf. Theor. 24, 530–536. https://doi.org/10.1109/TIT.1978.1055934.

Zhai, J., Li, Z., Wu, F., Xie, H., Zou, B., 2017. Quality assessment method for linear feature simplification based on multi-scale spatial uncertainty. ISPRS Int. J. Geo-Inf. 6 https://doi.org/10.3390/ijgi6060184.

Zhang, M., Kujala, P., Hirdaris, S., 2022. A machine learning method for the evaluation of ship grounding risk in real operational conditions. Reliab. Eng. Syst. Saf. 226 https://doi.org/10.1016/j.ress.2022.108697.

Zhang, S.K., Liu, Z.J., Cai, Y., Wu, Z.L., Shi, G.Y., 2016. AIS trajectories simplification and threshold determination. J. Navig. 69, 729–744. https://doi.org/10.1017/S0373463315000831.

Zhang, S.K., Shi, G.Y., Liu, Z.J., Zhao, Z.W., Wu, Z.L., 2018. Data-driven based automatic maritime routing from massive AIS trajectories in the face of disparity. Ocean Eng. 155, 240–250. https://doi.org/10.1016/j.oceaneng.2018.02.060.

Zhang, Y.Q., Shi, G.Y., Li, S., Zhang, S.K., 2020. Vessel trajectory online multi-dimensional simplification algorithm. J. Navig. 73, 342–363. https://doi.org/10.1017/S037346331900064X.

Zhao, L., Shi, G., 2018. A method for simplifying ship trajectory based on improved Douglas-Peucker algorithm. Ocean Eng. 166, 37–46. https://doi.org/10.1016/j.oceaneng.2018.08.005.

Zhao, L., Shi, G., 2019. A trajectory clustering method based on Douglas-Peucker compression and density for marine traffic pattern recognition. Ocean Eng. 172, 456–467. https://doi.org/10.1016/j.oceaneng.2018.12.019.

Zhong, Y., Kong, J., Zhang, J., Jiang, Y., Fan, X., Wang, Z., 2022. A trajectory data compression algorithm based on spatio-temporal characteristics. PeerJ Computer Science 8. https://doi.org/10.7717/peerj-cs.1112.

Zhou, Y., Daamen, W., Vellinga, T., Hoogendoorn, S., 2019. Review of maritime traffic models from vessel behavior modeling perspective. Transport. Res. C Emerg. Technol. 105, 323–345. https://doi.org/10.1016/j.trc.2019.06.004.

Zhou, Y., Daamen, W., Vellinga, T., Hoogendoorn, S.P., 2023a. Ship behavior during encounters in ports and waterways based on AIS data: from theoretical definitions to empirical findings. Ocean Eng. 272, 113879 https://doi.org/10.1016/j.oceaneng.2023.113879.

Zhou, Z., Zhang, Y., Yuan, X., Wang, H., 2023b. Compressing AIS trajectory data based on the multi-objective peak douglas–peucker algorithm. IEEE Access 11, 6802–6821. https://doi.org/10.1109/ACCESS.2023.3234121.

Zhu, F., Ma, Z., 2021. Ship trajectory online compression algorithm considering handling patterns. IEEE Access 9, 70182–70191. https://doi.org/10.1109/ACCESS.2021.3078642.