ISSN 2664-9969

UDC 004:656.61.052
DOI: 10.15587/2706-5448.2024.313523

Victoria Ponomaryova,
Pavlo Nosov

DEVELOPMENT OF A METHOD FOR PREDICTING HAZARDOUS SHIP TRAJECTORIES UNDER UNCERTAINTY OF NAVIGATOR ACTIONS

The object of the research is the automation processes in maritime navigation to ensure the safety of ship movement by predicting their trajectories in complex aquatic areas, such as narrow passages, straits, and ports. The research applied six key stages to create a comprehensive method for clustering and predicting ship trajectories based on ECDIS data.

In the first stage, ship movement trajectories were constructed according to risk categories, using the LCSS and DTW algorithms to compare planned and actual trajectories. This allowed for the accurate identification of course deviations and the determination of potentially dangerous sections of the trajectory. The second stage implemented clustering using the DBSCAN and GMM algorithms. DBSCAN was used to identify the density of points in space, and GMM provided modeling of cluster probabilities, allowing for better risk zone determination. The third stage applied the Douglas-Peucker compression algorithm to reduce the number of points in the trajectories, which preserved key characteristics and optimized data processing. In the fourth stage, ship movement stability was assessed using the Fourier transform, which allowed the detection of high-frequency oscillations that may indicate movement instability caused by changes in course or speed. The fifth stage included fuzzy clustering of trajectories using the Gaussian Mixture Model (GMM), which allowed modeling the probabilities of dangerous trajectories, considering the uncertainty of navigational parameters. At the final stage, a multilayer neural network (MLP) was used to predict future points of ship trajectories. The model accurately predicted the ship's coordinates, enabling timely trajectory adjustments.

Experimental results showed that the developed method increased the accuracy of ship trajectory prediction to 72–81 % and also significantly reduced the final error, ensuring effective risk management during complex navigation.

Keywords: *maneuvering in confined waters, emergencies, operational reliability, optimization of control processes, steering control, automatic control module.*

Received date: 21.08.2024
Accepted date: 15.10.2024
Published date: 23.10.2024

How to cite
Ponomaryova, V., Nosov, P. (2024). Development of a method for predicting hazardous ship trajectories under uncertainty of navigator actions. Technology Audit and Production Reserves, 5 (2 (79)), 44–55. https://doi.org/10.15587/2706-5448.2024.313523

1. Introduction

Modern maritime navigation faces numerous challenges, among which the most important is ensuring the safety of ship movement and effective planning of their trajectories in conditions of complex water routes, particularly in narrow passages, straits, and ports [1]. Automated navigation systems (ECDIS, AIS, ARPA, GPS, etc.) significantly improve these processes' safety and reliability. However, the complexity of processing large amounts of data, especially considering the experience and qualification characteristics of the navigator [2, 3], and the necessity for accurate real-time prediction of ship trajectories require new approaches to clustering, prediction, and data analysis.

Considering the above, it is proposed that the issue be investigated by critically analyzing current literature sources that describe studies related to the article's topic.

In work [4], a detailed review of methods for clustering ship trajectories is provided, including similarity measurement and clustering algorithms (DBSCAN, GMM) and data preprocessing methods such as reconstruction, compression, and segmentation. The authors emphasize the effectiveness of the Douglas-Peucker algorithm for trajectory compression, which significantly increases clustering accuracy. Such methods cover ship movement prediction, anomaly detection, and route optimization. Among the advantages is the detailed experimental evaluation of algorithms; however, the work needs to include more coverage of the latest neural networks for clustering and the absence of actual examples for emergency response.

The authors of work [5] propose a new algorithm for compressing ship trajectories that preserves the direction of movement based on the Open Window algorithm. The authors address the problem of excessive AIS (Automatic Identification System) data by compressing ship trajectories while preserving key direction change points, particularly for online and offline modes. A significant advantage is reducing trajectory compression time by 87.3 % for port

data compared to distance-based algorithms. Experiments significantly improve when applying this method for clustering tasks, anomaly detection, and ship trajectory prediction. A drawback is that the algorithm cannot guarantee global optimality due to its heuristic nature.

In the research [6], the problem of predicting ship trajectories under heterogeneous and multimodal movement patterns is considered, emphasizing the importance of accurate prediction to prevent collisions at sea. The authors propose a new approach to clustering ships based on historical AIS data, which allows for improving prediction accuracy using machine learning. Experiments showed a significant reduction in forecast errors (370 meters for a horizon of 10 minutes) when using the Random Forest algorithm. The main advantage is the reduction of computational complexity by using smaller data sets containing similar trajectories. A drawback of the approach is the complexity of clustering in places where routes intersect or overlap, which may reduce prediction accuracy for some ships.

A method for predicting ship trajectories [7] is proposed based on the use of graph spatiotemporal neural networks (ASTGCN) in combination with a correlation matrix obtained using StemGNN. The model accounts for nonlinear spatiotemporal dependencies in large AIS (Automatic Identification System) data sets, which allows for improving prediction accuracy. Experimental results showed that ASTGCN has more minor errors compared to other models, such as CNN_LSTM_CBAM and TCN, demonstrating high prediction accuracy. However, the model has increased computational complexity due to spatiotemporal attention and an adaptive adjacency matrix, which may require optimization for working with large volumes of data.

Another model [8] uses multi-level characteristics of ships (Multi-Rep), which include both surface attributes (length, width, draft) and deep characteristics (preferences in places of sailing and time). A feature fusion module (FFDM) is applied to predict future trajectories, which integrates trajectory data with ship characteristics. Experimental results showed that VEPO-S2S outperforms other models in both quantitative and qualitative indicators. However, the model also has high computational complexity due to the integration of multilevel characteristics, which may require optimization for working with large data sets.

In [9], a new learning model CLAIS is presented for computing the similarity of ship trajectories based on graph neural networks (GNN). The main goal of CLAIS is to solve the problem of limited learning methods for computing trajectory similarity in water transport, mainly due to the absence of data labels and spatial modeling. CLAIS uses graph learning to model spatial dependencies of trajectories and introduces a parameterized trajectory extension scheme. The model demonstrates high-efficiency thanks to learned similar samples and improved contrastive loss. However, the model faces several significant drawbacks. Firstly, it has high computational complexity due to graph neural networks, which increases resource requirements when working with large data sets. Secondly, the complexity lies in scaling the model to accurate navigation data, especially in cases with gaps or data distortion due to limited AIS signal quality.

The work [10] presents a method for analyzing complex trajectories of fishing vessels using the Fourier transform for identifying movement characteristics. The authors use the Fourier transform and clustering based on Gaussian mixtures to solve problems of uncertainty and complexity of fishing vessel trajectories. Essential results of the research are the identification of behavioral characteristics of vessels and the determination of zones of uncertain movement with a probability of 80 % within 1,000 meters and 50 % within 2,000 meters. This research is very important for fishing regulatory authorities to ensure safe navigation. The main advantages are an innovative approach to processing complex trajectories through the Fourier transform and effective clustering to determine danger zones. However, the work has certain limitations, particularly in the complexity of scaling this approach for other types of vessels and trajectories and in computational costs for processing large data arrays, which may complicate its use in actual maritime navigation conditions.

A hybrid approach to predicting ship trajectories [11] is presented, combining graph attention neural networks (GAT) and long short-term memory (LSTM). The GAT-LSTM model considers both spatial and temporal features, significantly improving trajectory prediction accuracy. Using AIS data for the port of Xiamen shows that this approach outperforms traditional models, reducing mean error by 44.52 % and final error by 56.20 %. The main advantages are improved accuracy when predicting complex situations, such as collision avoidance, and dynamic determination of the weight of neighboring ships affecting the target ship. However, the model has certain drawbacks, including high computational complexity due to the need to process a large amount of AIS data and difficulty scaling the model for a more significant number of ships in complex navigation environments.

In work [12], a new algorithm for compressing ship trajectories VATDC_CCRI is presented, which considers critical regions of the trajectory and uses the Douglas-Peucker algorithm together with a "sliding window" for efficient compression of AIS data. The main advantage is the ability to preserve essential points of course and speed changes of the ship, minimizing information loss, improving prediction accuracy and processing large data arrays. Experiments showed that VATDC_CCRI outperforms classical algorithms, providing higher processing speeds and better compression without significant loss of accuracy. The main drawbacks are high computational complexity due to the necessity of multiple iterations to determine critical regions and difficulties in scaling the algorithm for large data sets, which complicates its application in natural navigation systems.

Also, a new approach to predicting ship trajectories on inland waterways [13] has been studied, combining Gaussian mixture models (GMM) with transformers. The main idea is to integrate additional data, such as water discharge and geometry of waterways, to improve prediction accuracy. Research results show that this approach significantly improves prediction accuracy compared to previous models, such as N-CSCT, mainly due to considering the navigation context. GMM allows accounting for multimodal distributions of ship speeds and their positions relative to the fairway, which increases prediction accuracy under different water discharge conditions. However, drawbacks include considering large hydrological data, which may complicate scaling the model to other regions where such data may be absent or insufficiently accurate.

A method for predicting ship trajectories based on AIS data [14] is proposed, using bidirectional long short-term memory (Bi-LSTM). The main goal of the research is to improve prediction accuracy by cleaning data (removing

ISSN 2664-9969

anomalies and normalization) and applying Bi-LSTM to enhance forecast quality. Experiments showed that the Bi-LSTM method demonstrates lower mean absolute error (MAE), mean absolute percentage error (MAPE), and root mean square error (RMSE) compared to other models (ETS, ARIMA, SVR, RNN, LSTM). A drawback is the necessity to consider that accurate AIS data often have significant errors, particularly gaps or unstable signals due to bad weather conditions or technical problems. Also, it is only sometimes possible to guarantee that all anomalies will be processed appropriately, which may affect prediction accuracy.

In work [15], the problem of controlling the trajectory of an autonomous ship under unknown dynamic parameters and disturbances is considered. The authors propose an approach based on a proportional-integral-derivative (PID) controller with an adaptive sliding surface to solve the problem of tracking a ship's trajectory moving at low speed. The main advantage of the proposed method is its ability to ensure system stability through the use of Lyapunov theory and saturation functions for stability analysis. The model's drawbacks are its complexity in tuning due to the necessity for precise measurement of the ship's state and high resource requirements for accounting for unknown parameters and external disturbances.

The authors of work [16] present a new trajectory tracking scheme for underactuated ships with nonlinear dynamics with non-minimum phase behavior. The authors use coordinate transformation to convert the ship model into a strict feedback form with two input and two output signals, allowing backstepping control methods to be used. The main innovation is applying the concept of asymptotic modification of orientation (AMO), which helps avoiding singularities during recursive control design. The drawbacks are that the proposed method is complex for application in natural conditions due to the necessity for accurate ship dynamic parameters for systems with many external disturbances, such as wind or waves.

In work [17], a new approach to clustering ship trajectories based on AIS data is described, using a combination of the Douglas-Peucker (DP) algorithm for trajectory compression, Longest Common Subsequence (LCSS) for similarity measurement, and DBSCAN for clustering. The main goal is to accelerate the similarity measurement process and improve the accuracy of ship clustering in tight waterways. However, the main difficulties lie in the high computational costs that arise while processing large volumes of AIS data, as well as the need to adjust parameters for each of the algorithms, which complicates the automation of the process and scaling for different water zones.

In [18], an improved ship trajectory prediction model based on multilayer bidirectional recurrent neural networks with GRU blocks (Stacked-BiGRUs) is presented. The authors integrated several innovations, including an algorithm for removing anchored trajectories and restoring abnormal points and a two-stage clustering algorithm D-KMEANS for classifying ship behavior. Thanks to this, the model significantly reduced mean squared error (MSE) and mean absolute error (MAE) by 27 % and 46 %, respectively, indicating its high efficiency. The main advantages of the model are its ability to work under conditions of high traffic density and improved prediction in complex navigation conditions. However, problems may arise when working with abnormal trajectories or ships with different behavior patterns, which complicates the scaling and generalization of the model for different navigation conditions.

Thus, based on the analysis, it can be assumed that existing clustering and predicting ship trajectories, mainly based on AIS and ECDIS data, have significant advantages, including high prediction accuracy, efficient data compression, and the ability to detect anomalies. However, many of these approaches face problems of high computational complexity, insufficient optimization for working with large data volumes, and difficulties in scaling models for different navigation conditions. Other limitations include the complexity of clustering at route intersections, processing abnormal trajectories, data gaps or distortions, and adapting to navigator actions under risk conditions, i. e., considering the human factor. This indicates the need to develop comprehensive solutions that combine modern clustering algorithms with machine learning methods for more efficient and adaptive data processing.

The research aims to develop and implement a comprehensive method, whose stages will involve clustering and predicting ship trajectories based on ECDIS data to ensure navigation safety in narrow passages and complex aquatic areas.

The main task of the research is to develop software that will work in online mode and, according to the algorithms of the developed comprehensive method, will have the following sequential functionality capable of:

– constructing ship movement trajectories according to risk categories by analyzing ECDIS data to assess risk based on distance to the shoreline;
– performing clustering of ship trajectories using DBSCAN and D-KMEANS, relying on algorithms for classifying ship behavior based on historical data and trajectory clustering;
– using the trajectory compression procedure using the Douglas-Peucker algorithm to improve clustering efficiency and reduce computational costs;
– assessing ship movement stability using Fourier analysis by identifying characteristic trajectory oscillations to improve prediction accuracy;
– performing fuzzy clustering of trajectories using fuzzy logic to assess the degree of risk and classify dangerous situations;
– predicting ship trajectories using neural networks in two possible directions. Analyzing ship movement data to optimize future trajectories and improve prediction accuracy.

2. Materials and Methods

The object of the research is the processes of automation in maritime navigation aimed at ensuring the safety of ship movement by predicting their trajectories in complex aquatic areas (narrow passages, straits, ports). Particular attention is paid to automated ship control and monitoring systems such as ECDIS, AIS, ARPA, and GPS, which are used for collecting and analyzing navigation data, as well as factors influencing the navigator's decision-making under risk conditions.

The research focuses on developing a comprehensive method that includes clustering, compression, and prediction of ship trajectories to improve the accuracy and speed of data processing, considering the human factor, the navigator's qualifications, and ensuring effective voyage planning.

A systematic approach was used in the research, along with data analysis and synthesis methods. Clustering methods (DBSCAN, D-KMEANS) were applied for processing and analyzing ship trajectories, numerical integration for assessing movement stability using Fourier analysis, and prediction methods using neural networks (multilayer

perceptron (MLP) with subsequent activation through the ReLU function). A personal computer with a Windows 10 operating system, Anaconda environment, and Python were used for implementation. The analysis software included modules for processing ECDIS data of the Bosporus Strait and custom-developed algorithms for clustering, trajectory compression, and prediction based on navigation system data. This allows working with large volumes of data and ensures accuracy in risk classification and dangerous situation prediction using an interactive web map in online mode.

The development of the method involved performing a series of sequential research tasks through mathematical description and software implementation, which will be described in this section. Additionally, existing modeling results will be presented as screenshots of the interactive web map.

The initial operation was creating and selecting reports using the ECDIS TRANSAS NTPRO 4000/5000 server, which stores current ship movement data and navigation data at intervals of 5 seconds. Then, based on a control report optimized to CSV format, mathematical processing of big data was conducted using the developed software in Python.The stages of the method and the corresponding points are outlined below.

2.1. Construction of ship movement trajectories according to risk categories

2.1.1. Application of the LCSS algorithm to determine similarity between the actual and reference trajectory. This is done using the formula:

$$LCSS(T_1,T_2)=\max\begin{pmatrix} LCSS(T_1-\{t_{1_i}\},T_2),\\ LCSS(T_1,T_2-\{t_{2_j}\}),\\ LCSS(T_1-\{t_{1_i}\},T_2-\{t_{2_j}\})+1 \end{pmatrix}, \quad (1)$$

where $LCSS(T_1, T_2)$ is the metric that measures similarity between two trajectories T_1 and T_2, t_{1i}, t_{2j} are points on trajectories T_1 and T_2, respectively, which can be ignored if they deviate insignificantly.

When comparing the planned ship movement trajectory with the control one, there are two trajectories $T_1 = \{t1_1, t1_2,..., t1_m\}$ and $T_2 = \{t2_1, t2_2,..., t2_n\}$, where $t1_i$ and $t2_j$ represent the coordinates of points on trajectories T_1 and T_2, respectively.

This allows the LCSS algorithm to determine the length of the longest common subsequence between T_1 and T_2 using the following recursive formula:

$$LCSS(i,j)=\begin{cases} 0, & \text{if } i=0 \text{ or } j=0,\\ LCSS(i-1,j-1)+1, & \text{if } |t_{1_i}-t_{2_j}|\le \delta i\,|\,t_{1_i}^{time}-1,\\ \max(LCSS(i-1,j),LCSS(i,j-1)), & \text{else,}\end{cases} \quad (2)$$

where $LCSS(i, j)$ is the length of the longest common subsequence between the first i points of trajectory T_1 and the first j points of trajectory T_2; t_{1i}, t_{2j} are the coordinates of the i-th point on trajectory T_1 and the j-th point on trajectory T_2, respectively; δ is the allowable spatial-temporal distance between points t_{1i} and t_{2j}, which allows for considering deviations.

2.1.2. Dynamic time warping (DTW), accounting for time shifts when comparing trajectories:

$$DTW(T_1,T_2)=\min_{\pi}\sum_{k=1}^{K} d\left(t_{1_{\pi_1(k)}},t_{2_{\pi_2(k)}}\right), \quad (3)$$

where $DTW(T_1, T_2)$ is the metric that measures the difference between trajectories T_1 and T_2, considering time shifts; $t_{1_{\pi_1(k)}}, t_{2_{\pi_2(k)}}$ are points on the trajectories that are aligned with each other using π (permutation); $d(t_{1_{\pi_1(k)}}, t_{2_{\pi_2(k)}})$ is the distance function between points $t_{1_{\pi_1(k)}}, t_{2_{\pi_2(k)}}$; π is the optimal alignment of points on the trajectories; K is the number of steps in the alignment (path).

2.2. Clustering of ship trajectories using DBSCAN and K-MEANS

2.2.1. DBSCAN clustering algorithm (Density-based spatial clustering of applications with noise). The DBSCAN algorithm [19] performs clustering based on the density of points in space, using two main parameters: neighborhood radius (ε) and minimum number of points (*MinPts*).

Main parameters of the algorithm:

– ε – *neighborhood radius*: the maximum distance between two points for one point to be considered a neighbor of the other;

– *MinPts – minimum number of points*: the minimum number of points that must be within the radius ε for a point to be considered a "core" point and become part of a cluster.

For calculations, it is necessary to operate with the following parameters:

The distance between two points is chosen depending on the situation (Euclidean distance, Manhattan distance, or Chebyshev distance).

The neighborhood of point p is defined as the set of all points located at a distance not greater than ε from point p:

$$N_\varepsilon(p)=\{q\,|\,dist(p,q)\le\varepsilon\}, \quad (4)$$

where $dist(p, q)$ is the selected distance metric.

Core point: A point p is a core point if the number of points in its neighborhood $N\varepsilon(p)$ exceeds or equals *MinPts*:

$$|N_\varepsilon(p)|\ge MinPts. \quad (5)$$

Border point: A point q is a border point if it is in the neighborhood of a core point but is not itself a core point.

Noise point: A point p that is neither a core nor a border point is considered noise:

$$p\notin cluster,\ p\notin N_\varepsilon(q)\ for\ all\ q. \quad (6)$$

2.2.2. D-KMEANS algorithm combining DBSCAN and K-Means for better trajectory clustering. DBSCAN is used to identify initial clusters, and K-Means is used to compute the centroid μ_k of the k-th cluster:

$$\mu_k=\frac{1}{|C_k|}\sum_{x_i\in C_k} x_i, \quad (7)$$

where C_k is the set of points belonging to the k-th cluster; x_i is the coordinates of the i-th observation point; μ_k is the centroid of the k-th cluster.

ISSN 2664-9969

Cluster updating procedure: After initial cluster formation by DBSCAN, K-Means refines the boundaries by minimizing the sum of squared distances from each point to the nearest centroid:

$$\arg\min_C \sum_{k=1}^{K} \sum_{x_i \in C_k} dist\left(x_i, \mu_k\right)^2, \quad (8)$$

where C is the current distribution of points among clusters; μ_k is the centroid of the k-th cluster; $dist(x_i, \mu_k)$ is the distance from the end to the centroid.

Thus, the D-KMEANS algorithm allows a more accurate classification of ship trajectories into groups based on their behavior. This can help identify trajectories with increased risk, focus on the most important routes, and, if necessary, take measures for their correction.

2.3. Compression of trajectories using the Douglas-Peucker algorithm without losing key characteristics

2.3.1. Distance from a point to a line. Let $P_1(x_1, y_1)$ and $P_n(x_n, y_n)$ be the first and last points of the trajectory, and $P_i(x_i, y_i)$ be any other point of the trajectory. The distance d_i from point P_i to the line connecting P_1 and P_n is calculated using the formula:

$$d_i = \frac{\left|(y_n - y_1)x_i - (x_n - x_1)y_i + x_n y_1 - y_n x_1\right|}{\sqrt{(y_n - y_1)^2 + (x_n - x_1)^2}}. \quad (9)$$

2.3.2. Selection of the point with maximum distance d_{max}. Determine the point P_k with the maximum distance:

$$d_{\max} = \max_{1<i<n} d_i.$$

2.3.3. Threshold value ε. If $d_{\max} > \varepsilon$, the point P_k, where $k = \text{argmax}_1 < i < n d_i$, remains in the trajectory, and the algorithm is recursively applied to two parts of the trajectory: from P_1 to P_k and from P_k to P_n.

The Douglas-Peucker algorithm [20] effectively reduces the number of points in a trajectory while preserving its main characteristics. This reduces the amount of data for further processing and analysis, facilitates visualization, and preserves essential information about the shape of the ship's trajectory.

2.4. Assessment of ship movement stability using Fourier analysis

2.4.1. Fourier transform for analyzing frequency Components of oscillations to assess movement stability. Analyzing the stability of a ship's trajectory is critical for ensuring navigational safety. Fluctuations in course, speed, and acceleration can indicate instability arising from insufficient qualification or experience in operating the ship. The application of the Fourier transform [21] allows for the detection of the frequency components of these fluctuations and the assessing their impact on the overall stability of the trajectory. To enhance the accuracy and reliability of the analysis, additional aspects are added to the model, such as noise filtering, nonlinear analysis, correlation analysis, and others.

Let's describe the sequence of mathematical calculations for this stage.

2.4.1.1. Temporal trajectory. For each ship, there is a set of data that includes the following variables: Time (s) – time interval t; Speed – ship speed $v(t)$; Course – ship course $\theta(t)$; Latitude and Longitude – coordinates of the ship's trajectory $x(t)$, $y(t)$; Mamdani Acceleration, Sugeno Acceleration, ANFIS Acceleration – accelerations calculated by different methods.

The ship's trajectory can be represented as a complex signal in the time domain:

$$z(t) = x(t) + jy(t).$$

2.4.1.2. Fourier transform. To convert the time signal into the frequency domain, the discrete Fourier transform is used:

$$Z(f) = \sum_{n=0}^{N-1} z(n) e^{-j2\pi fn/N}, \quad (10)$$

where $Z(f)$ is the frequency spectrum of the signal $z(t)$, showing the intensity of oscillations at different frequencies f.

2.4.1.3. Spectrum analysis. The spectrum of the signal $Z(f)$ contains information about the frequency components of the ship's course and speed:

$$z(t) = \sum_{k=1}^{N} A_k \cos\left(2\pi f_k t + \phi_k\right), \quad (11)$$

where A_k is the amplitude of oscillations, f_k is the frequency of oscillations, and ϕ_k is the phase shift.

2.4.1.4. Low-frequency filtering. To remove low-frequency noise that is not associated with control instability, a Butterworth filter is used:

$$H(f) = \frac{1}{\sqrt{1 + \left(\frac{f}{f_c}\right)^{2n}}}, \quad (12)$$

where f_c is the cutoff frequency; n is the filter order.

2.4.1.5. Stability assessment. To assess the level of instability, it is possible to calculate the mean energy of the high-frequency components:

$$E_{high} = \frac{1}{N} \sum_{k=1}^{N_{high}} \left|Z(f_k)\right|^2. \quad (13)$$

The qualification coefficient Q is defined as:

$$Q = \frac{1}{E_{high}}. \quad (14)$$

A low value of Q indicates a high level of instability, which may indicate insufficient navigator qualification.

This stage of the method allows for a comprehensive analysis of the ship's trajectory stability, taking into account both the main frequency components and additional aspects such as external factors and nonlinear oscillations.

2.5. Fuzzy clustering of trajectories using fuzzy logic

2.5.1. Gaussian mixture models (GMM). GMM models data as a combination of several Gaussian distributions:

$$p(x) = \sum_{k=1}^{K} \pi_k N\left(x \mid \mu_k, \Sigma_k\right),$$

$$N\left(x \mid \mu_k \Sigma_k\right) =$$

$$= \frac{1}{\sqrt{(2\pi)^d \left|\Sigma_k\right|}} \exp\left(-\frac{1}{2}(x - \mu_k)^T \Sigma_{k^{-1}} (x - \mu_k)\right), \quad (15)$$

where $p(x)$ is the probability that point x belongs to a certain mixture component; π_k is the weight of the k-th mixture component; $N(x|\mu_k, \Sigma k)$ is a Gaussian distribution with mean μ_k and covariance matrix Σk; K is the number of mixture components; μ_k is the mean value for the k-th component; Σk is the covariance matrix for the k-th component; d is the number of dimensions (space dimensionality).

The GMM algorithm [22] will allow modeling the probability of the ship being in a dangerous zone, taking into account variable navigation parameters. This helps predict risks associated with approaching the shoreline or other objects.

2.5.2. Expectation-maximization (EM) algorithm for GMM. The EM algorithm consists of two main stages that repeat until convergence.

2.5.2.1. Expectation step (E-step). For each observation x_i, the probability that it belongs to each mixture component k is calculated:

$$\gamma_{ik} = \frac{\pi_k \cdot N(x_i|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \cdot N(x_i|\mu_j, \Sigma_j)}, \quad (16)$$

where γ_{ik} is the responsibility of component k for point x_i.

2.5.2.2. Maximization step (M-step). The model parameters are updated based on the calculated responsibilities:

– Updating the weight:

$$\pi_k = \frac{1}{N} \sum_{i=1}^{N} \gamma_{ik};$$

– Updating the mean value:

$$\mu_k = \frac{\sum_{i=1}^{N} \gamma_{ik} x_i}{\sum_{i=1}^{N} \gamma_{ik}};$$

– Updating the covariance matrix:

$$\Sigma_k = \frac{\sum_{i=1}^{N} \gamma_{ik} (x_i - \mu_k)(x_i - \mu_k)^T}{\sum_{i=1}^{N} \gamma_{ik}}.$$

The algorithm repeats until the parameter changes become insignificant.

The Gaussian Mixture Model (GMM) will allow assessing risks in maritime navigation, considering a multitude of parameters.

2.6. Prediction of ship trajectories using a multilayer perceptron (MLP) model with subsequent activation through the ReLU function

Prediction stages:

1. *Dense layers*: The prediction model uses several fully connected (Dense) layers, where each layer applies a linear combination of input data.

2. *ReLU activation function* [23]: Each layer applies the ReLU (Rectified Linear Unit) activation function, which cuts off negative values, especially useful for predicting physical processes such as ship trajectories. This allows the network to consider only positive changes in parameters such as speed or course.

3. *Dropout to prevent overfitting*: To avoid overfitting, the model uses Dropout – randomly turning off neurons during training. This allows the model to avoid excessive adaptation to noise or features of the training data set.

4. *Data normalization*: To ensure that ship parameters such as coordinates and speed are on the same scale, normalization is applied through BatchNormalization(). This allows stabilizing the training and accelerating its convergence.

5. *Trajectory prediction*: The model uses the output layer to predict future values of the ship's coordinates.

6. *Loss function*: For training the model, the mean squared error (MSE) loss function is used, which minimizes the distance between the predicted and actual ship trajectory.

This model allows predicting deviations from the ship's course, taking into account current and previous movement parameters such as coordinates, speed, and course. The model is optimized for working with large data volumes due to the application of Dropout and normalization, which prevents overfitting and improves training convergence. The loss function based on mean squared error minimizes the difference between the actual and predicted trajectory, which is critically important for accurate modeling and enhancing navigational safety.

3. Results and Discussion

3.1. Development of software corresponding to Stage 2.1, describing on an interactive map the ship's movement trajectory and the Bosporus shoreline trajectory

Thanks to the developed software, the ship's movement trajectory and the Bosporus shoreline trajectory are compared based on the proximity of their coordinates, and, according to this criterion, the risk of collision of the ship with the contours of the strait's shoreline is determined.

3.1.1. Data loading and preparation. Data on ship trajectories and the shoreline are loaded from Excel files using pandas. The method *to_numpy()* converts the data into arrays for convenient computations.

3.1.2. Calculating the minimum distance to the shoreline. For each point of the ship's trajectory, the minimum distance to the shoreline is calculated using *geodesic()* from the geopy library. The program iterates through each point of the trajectory in a loop and stores the minimum distance value.

3.1.3. Updating risk levels. Depending on the obtained distances, the program assigns risk levels for each point of the trajectory. An *if* conditional statement is used, which distributes risk levels (e. g., "Critical", "Safe") depending on the distance to the shore.

3.1.4. Saving data. The updated data are saved to a CSV Excel file using the *to_excel()* method.

3.1.5. LCSS algorithm. The LCSS algorithm uses dynamic programming. A matrix *dp* is created, where for each pair of points of the two trajectories, their correspondence is checked according to the parameter *delta*.

3.1.6. DTW algorithm. The DTW algorithm creates a matrix filled with minimal alignment options for trajectory points. The cost of alignment is determined as the Euclidean distance between points.

3.1.7. Visualization on the map. Visualization of risks on the map is performed using the folium library. Points are displayed as markers with corresponding colors depending on the risk level in HTML format (Fig. 1).

ISSN 2664-9969

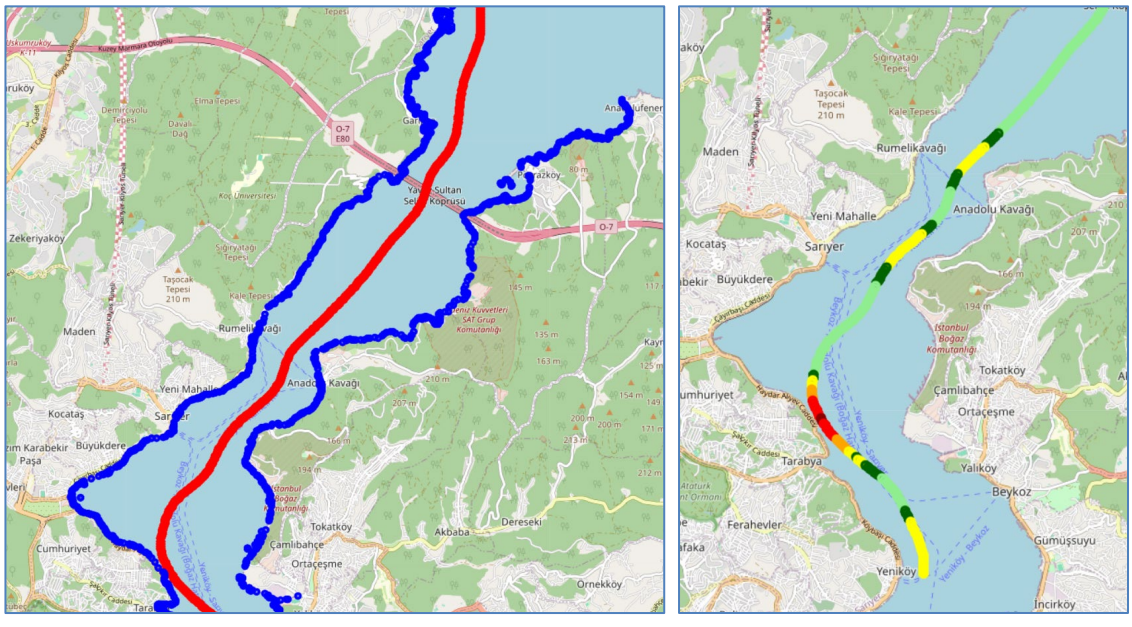

Fig. 1. Visualization of the movement trajectory, considering the proximity to the shoreline as a risk criterion

3.2. Development of software corresponding to Stage 2.2

It involves the following automation steps:

3.2.1. Data loading and preparation. The program loads ship trajectory data from an Excel file using pandas. Risk-level data are converted to numerical values using the *map()* method to ensure their further processing. This allows clustering to be applied based on the ship's speed, risk level, and minimum distance to the shoreline.

3.2.2. DBSCAN algorithm. DBSCAN detects clusters based on density. The main parameters are the radius ε and the minimum number of points *MinPts*. All data are scaled before clustering using *StandardScaler()*. The DBSCAN algorithm, implemented via the *DBSCAN()* function, classifies trajectory points into clusters based on Euclidean distance. The clustering result is stored in the variable *dbscan_clusters*.

3.2.3. Gaussian mixture model (GMM) Algorithm. After executing the DBSCAN algorithm, the GaussianMixture() (GMM) model is used for additional refinement of clusters. This clustering method allows evaluating the probability of points belonging to a specific cluster. The number of components (clusters) is set to 9, and probabilistic clustering is performed via *fit_predict()*. The result is stored in a new column *Cluster*.

3.2.4. Visualization of clusters on the map. To visualize clusters on an interactive map, the folium library is used. Each cluster is displayed in a specific color based on its value.

The program also adds a legend division for convenient display of clusters on the map. For each cluster, navigation tips and COLREG rules relevant to that cluster are output. The map is saved in HTML format.

3.2.5. D-KMEANS algorithm. DBSCAN is combined with K-Means for further refinement of clusters. First, DBSCAN forms initial clusters, and K-Means calculates centroids for each cluster, allowing minimizing the sum of squared distances from each point to the nearest centroid. This makes clustering more accurate, especially for complex trajectories. Centroids are calculated using formula (7), and the cluster updating procedure occurs by minimizing the sum of squared distances to the centroid (8).

3.2.6. Saving results. The updated report with clustering results is saved in Excel format using the *to_excel()* method (Fig. 2).

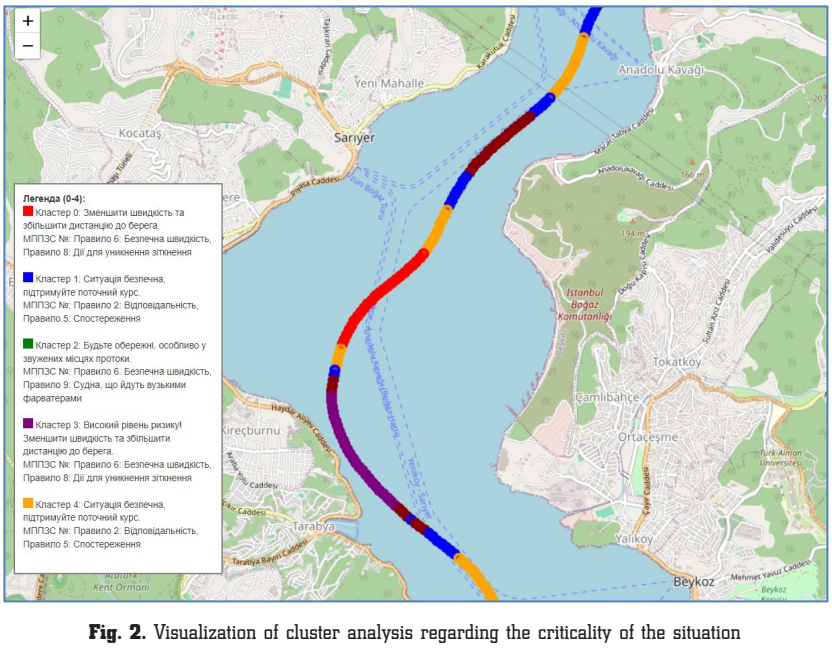

Fig. 2. Visualization of cluster analysis regarding the criticality of the situation

3.3. Algorithm of the developed program corresponding to Stage 2.3

3.3.1. Data loading. The program loads data using the *pd.read_excel()* function from the pandas library, which reads data from an Excel file. Data such as Latitude, Longitude, Speed, Risk Level, Min Distance to Shore (m), and Cluster

are converted into a DataFrame format. This provides a structure for further processing, where each row corresponds to a separate point of the ship's trajectory.

3.3.2. Douglas-Peucker algorithm for trajectory compression. The Douglas-Peucker algorithm, implemented via the *rdp* function from the rdp library, is used to reduce the number of points in the trajectory. The main parameter of the algorithm is *epsilon*, which defines the allowable distance between a point and the segment connecting two other trajectory points. The algorithm recursively removes points if their distance to the line connecting the end points of the segment is less than *epsilon*.

3.3.3. Saving compressed coordinates. After compression, the coordinates are saved into a new DataFrame using the *pd.DataFrame()* function. Other attributes of the points, such as Speed, Risk Level, Min Distance to Shore (m), and Cluster, are copied from the original data set via the *loc[]* method. This ensures that compressed trajectories retain all important metadata associated with each point.

3.3.4. Removal of anchored trajectories. To filter out "anchored" points (points with low speed), a conditional operator is used with a threshold value of Speed. Using the *loc[]* method, points with speed less than the set threshold are removed from the DataFrame. This reduces the amount of unnecessary data and helps focus on the main movements of the ship.

3.3.5. Ordering points by proximity. The *geodesic()* function from the geopy library is used to calculate geodesic distances between points, taking into account the curvature of the Earth's surface. To order points by their proximity, a procedure is applied where each point is compared with others to determine the nearest point. The nearest points are added to a list of ordered points, ensuring a logical sequence of the ship's movement.

3.3.6. Visualization of the compressed trajectory. For trajectory visualization, the folium library is used. The *folium.Map()* function creates an interactive map where the center of the map is defined by the mean value of the trajectory points' coordinates. Trajectory points are added to the map as markers using the *folium.CircleMarker()* method, where each point is marked with a color corresponding to its cluster. Straight lines between points are added using *folium.PolyLine()*.

3.3.7. Saving results. The results of compression and trajectory cleaning are saved into a new Excel file using the *to_excel()* function from pandas. The interactive map is saved in HTML format using the *save()* function from folium. This allows saving both the data for further analysis and the trajectory visualization for future use or viewing (Fig. 3).

3.4. Algorithm of the developed program corresponding to Stage 2.4

This algorithm operates as follows:

3.4.1. Data loading and preparation. Ship trajectory data are loaded from two Excel files using the *pd.read_excel()* method from pandas. One file contains compressed and cleaned ship trajectories, and the other contains risk data. After loading, the data are stored in DataFrame format for further analysis. In particular, ship coordinates are presented as Latitude and Longitude and additional data such as ship speed (Speed).

3.4.2. Coordinate transformation using multidimensional scaling (MDS). To simplify spatial analysis, Multidimensional Scaling (MDS) is used via the *MDS()* method from sklearn. This transformation reduces the dimensionality of the data from two spatial coordinates (latitude and longitude) to new MDS coordinates (MDS_X and MDS_Y), simplifying analysis and visualization.

3.4.3. Coordinate transformation into polar system. The coordinates obtained after MDS (MDS_X and MDS_Y) are transformed into a polar coordinate system to better represent and interpret the ship's trajectory. The radial distance *r* is calculated using the *np.sqrt()* function, and the angle *theta* is calculated using *np.arctan2()*. This allows for building polar plots of the ship's trajectory, adding a new dimension for visualization and analysis.

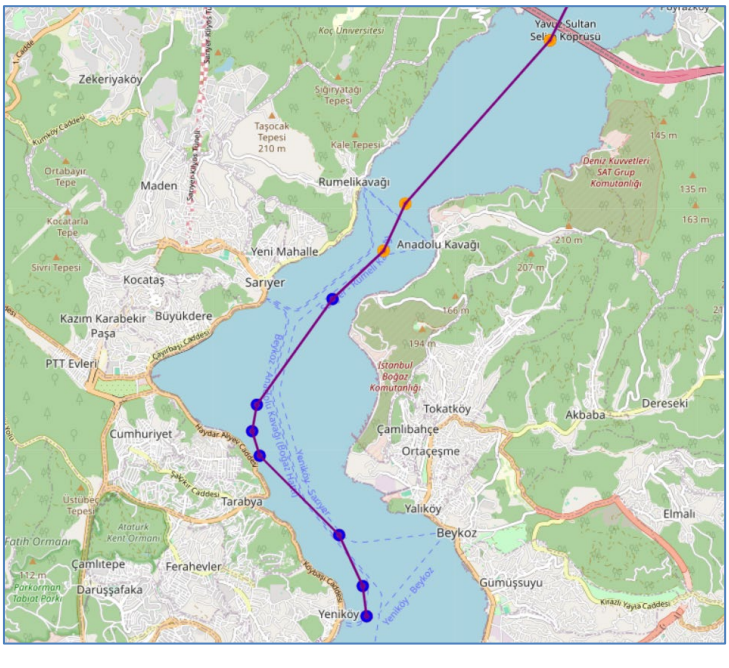

Fig. 3. Visualization of the trajectory compression function to accelerate real-time data processing

ISSN 2664-9969

3.4.4. Evaluation of distance to the shoreline. The distance to the shoreline is stored in the "Min Distance to Shore (m)" column to analyze the ship's movement stability. The program calculates the mean, maximum, and minimum values of this distance using the *mean()*, *max()*, and *min()* methods. This information is used to assess navigation safety.

3.4.5. Analysis of frequency components using Fourier transform. To analyze the frequency components of the ship's speed oscillations, the Discrete Fourier Transform (DFT) is used. This allows the conversion of ship speed data from the time domain to the frequency domain. Using the *np.fft.fft()* function from the numpy library, the frequency spectrum for speed is calculated, enabling visualization of which frequencies dominate in the ship's speed signal.

First, a frequency array is created using the *np.fft.fftfreq()* function, where the number of data points and the time interval between them are set. Then, the *np.fft.fft()* function is applied to the ship's speed data to obtain the frequency spectrum. Oscillation amplitudes are calculated via the modulus of complex numbers in the frequency spectrum using *np.abs()*. An amplitude spectrum plot is constructed to visualize the main frequency components affecting the ship's movement.

3.4.6. Filtering low-frequency noise. A Butterworth filter is used to remove low-frequency noise that may not be associated with the ship's control stability. This allows cleaning the signal from noise and focusing on the frequency components responsible for movement instability. The filter is configured through the cutoff frequency (*fc*) and filter order (*n*) parameters.

3.4.7. Stability assessment. The ship's movement stability is assessed based on the frequency spectrum.

High-frequency components may indicate movement instability caused by abrupt changes in course or speed. The energy of these oscillations is calculated by summing the squares of the amplitudes using the *np.sum()* function, allowing determination of the level of instability.

3.4.8. Calculation of amplitudes. Amplitudes are calculated via *np.abs()* to obtain the moduli of complex numbers and are normalized by multiplying by *2.0/N*.

3.4.9. Stability assessment. High-frequency components are extracted using *np.where()*, and the sum of their amplitudes is calculated using *np.sum()* to assess stability.

3.4.10. Evaluation of regularity and speed influence. Maximum amplitudes are calculated via *np.max()*, and the variation of speed is calculated via *np.var()*, providing regularity and speed influence estimates.

3.4.11. Visualization of results. The analysis results are presented in several plots (Fig. 4 and Fig. 5), built using the matplotlib library. Visualizations include: a plot of coordinates after MDS for spatial analysis (*plt.scatter()*); a polar plot of the ship's trajectory (*plt.polar()*); a plot of the change in distance to the shoreline to assess the trajectory relative to the shore (*plt.plot()*); and an amplitude spectrum of the ship's speed for analyzing frequency components (*plt.plot()*).

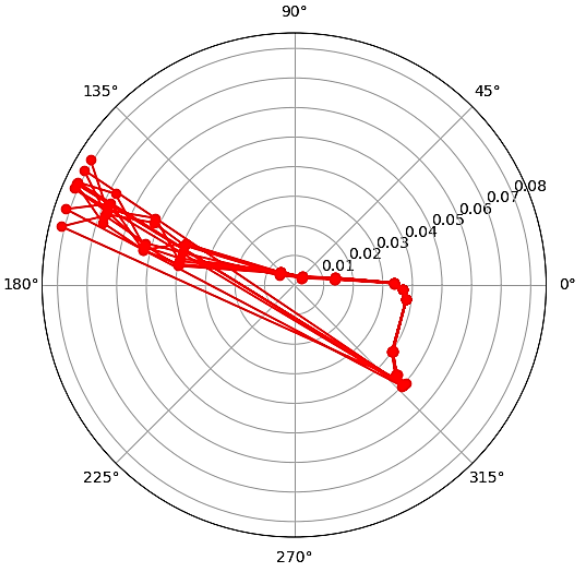

Fig. 4. Polar plot of the ship's movement trajectory

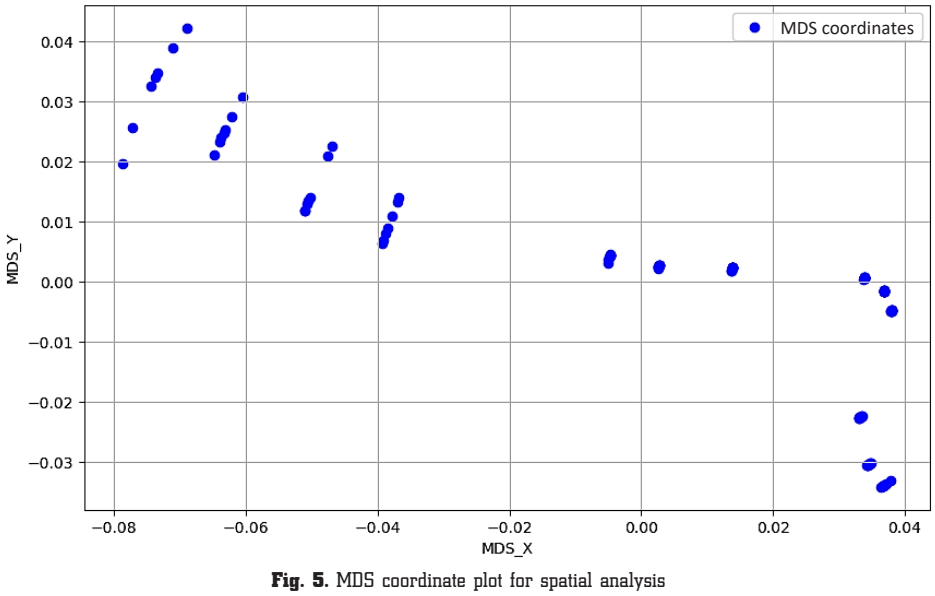

Fig. 5. MDS coordinate plot for spatial analysis

3.5. Automation elements corresponding to Stage 2.5

Gaussian Mixture Model (GMM): In the program code for clustering, K-Means is initially applied for the initial distribution of points into clusters using *KMeans(n_clusters=k).fit()*. After that, probability modeling is conducted using Gaussian Mixture Models (GMM), where each cluster is modeled as a multivariate normal distribution.

Data are standardized using *StandardScaler().fit_transform()*, allowing accurate probability modeling for each cluster. To calculate the probability density, the function *multivariate_normal(mean=μk, cov=Σk)* is used, which estimates the probability that a point belongs to a specific cluster based on the mean values μ*k* and covariance matrices Σ*k*.

EM Algorithm (Expectation-Maximization): The EM algorithm is applied to refine cluster parameters. The E-step

is implemented by calculating the responsibility for each point via *multivariate_normal.pdf()*. The M-step involves updating each cluster's mean values and covariance matrices, although this needs to be more detailed in the code. The initial clustering by K-Means helps quickly approach the solution.

Visualization of Results: For visualization, contour plots are used via *plt.contour()* to display cluster boundaries based on probability density and *plt.scatter()* to show the trajectory points themselves labeled by clusters. This allows for assessing the distribution of trajectories and possible dangerous zones.

At this stage, input data are created for further prediction using neural networks.

3.6. Automation using a multilayer neural network (MLP) corresponding to Stage 2.6

Data preparation: First, data are loaded from an Excel file, which includes ship trajectory parameters: latitude, longitude, speed, and course. Data are scaled using *StandardScaler()* to reach the same range.

MLP model: The model is built as a multilayer neural network (MLP) via the *Sequential()* class. It consists of several layers:

First layer: Dense(128, activation='relu'), uses 128 neurons and the ReLU activation function.

Normalization layers: Added via *BatchNormalization()*, which helps stabilize training.

Dropout layers with a probability of 0.2 (*Dropout(0.2)*): Prevent overfitting.

Final layer: Has two neurons for predicting the latitude and longitude of the next trajectory point.

Model training: The model is compiled using the *adam* optimizer and the *mean_squared_error* loss function, which minimizes the difference between actual and predicted points. Early stopping (*EarlyStopping*) is applied during training to avoid overfitting if the model does not improve for ten epochs.

Prediction: After training, the model uses input data, which are scaled, to predict the following trajectory points. Predictions are restored to the original scale via *scaler_y.inverse_transform()*.

Visualization: Prediction results are displayed on a map using the folium library (Fig. 6). Predicted points are marked with red markers, and the original trajectory is shown with a blue line.

As can be seen, using the method of predicting dangerous ship movement trajectories, it was possible to identify two possible forecast branches in the most dangerous zone of the route, where the distance to the shore was less than 100 meters. The safe forecast branch had three points, indicating that it is less probable than the dangerous one, which had over 10 points. Such a state of affairs forced a switch to emergency automatic control of the ship's movement using the automatic control module, which corrected the situation until the watch was reinforced and the ship's position in the strait became less critical.

Limitations of the Research. The approaches and methods proposed in the research have limitations that may affect the relative effectiveness of implementing the developed prediction method. Automated clustering algorithms (DBSCAN, D-KMEANS) and machine learning methods (MLP), which require significant computational power, complicate real-time operation when processing large volumes of ECDIS navigation data. The reduction of this factor's impact is achieved by applying the Douglas-Peucker algorithm for trajectory compression. Still, the effect of this approach does not exceed 24–27 % in terms of reducing total time.

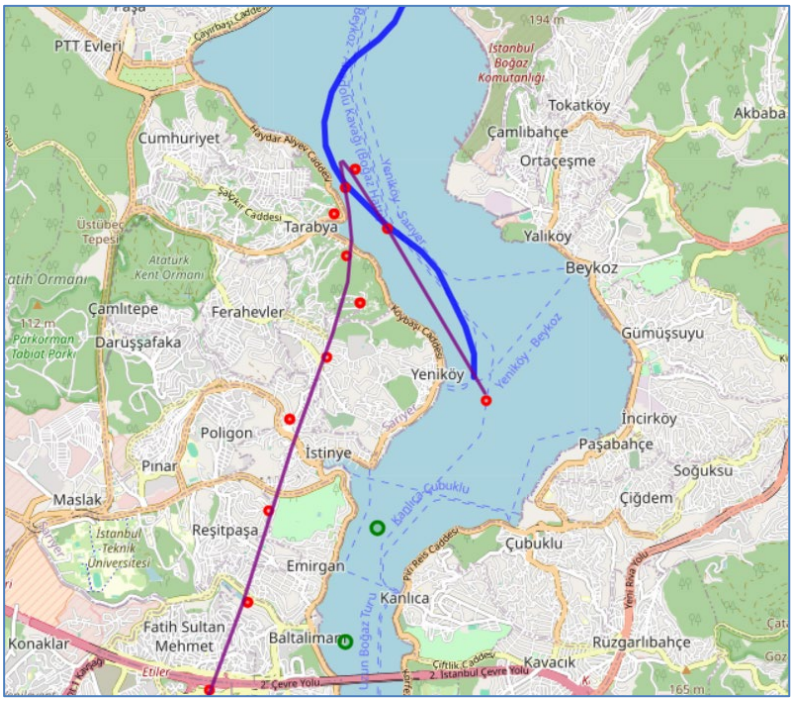

Fig. 6. Prediction of waypoints in a hazardous zone

Another limitation is the quality of input data, mainly if ECDIS display recognition is applied using computer vision methods in this process, which may contain gaps or anomalies requiring additional cleaning procedures to improve prediction accuracy.

Impact of Martial Law Conditions. One of the conditions of martial law is the necessity for Ukraine to maintain the export of products, mainly agricultural and partially industrial. The term "grain corridor" is associated with the safety of maritime transportation from the ports of Pivdennyi (Odesa) and Ochakiv (Mykolaiv). Clarity in the navigator's actions, readiness to avoid sudden danger zones from the aggressor country, and the ability to perform maneuvers to diverge from the trajectories of drones and ballistic missiles are necessities that require special skills (Fig. 7).

In Fig. 7, it can be seen that the navigator received warnings via communication means about zones not displayed on the ECDIS and radar screens, made notes using the Man Corr panel tools, and performed a divergence maneuver, orienting by the fairway boundaries and the values of safe Under Keel Clearance and Deep contour.

Prospects for Further Research. For further research, it is a priority to expand three key aspects: increasing the volume of data, improving prediction methods, and integrating real-time decision support mechanisms.

ISSN 2664-9969

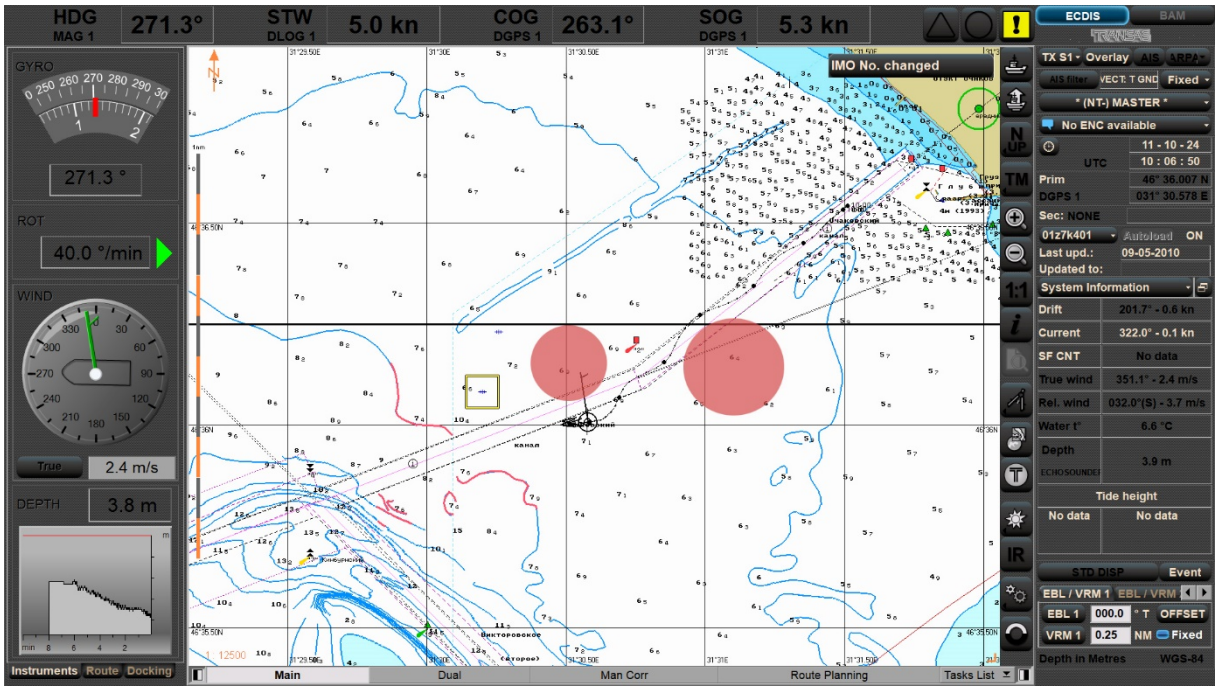

Fig. 7. Avoiding potential threats near Kinburn Spit

In particular, to achieve more accurate navigation, more information about sea conditions, depth fluctuations, and current dynamics should be added. Collecting historical data on ship trajectories in different weather conditions (data from company servers, IMO, IHO) is important, as it will help identify patterns of risky zones and typical navigator errors.

Additionally, machine learning algorithms such as recurrent neural networks or convolutional neural networks should be improved to create adaptive models that will consider variable navigation factors in real-time, particularly military threats. This will allow for faster responses to changes in the navigation situation and reduce the likelihood of navigator-operator errors in conditions of limited visibility and complex routes.

4. Conclusions

In this research, a comprehensive method for predicting dangerous ship trajectories was developed and applied, encompassing six key stages. Each of these stages was based on modern clustering algorithms, data analysis, and machine learning, which ensured effective resolution of the set tasks.

The first stage involved constructing ship movement trajectories according to risk categories, using the LCSS algorithm to compare planned and actual trajectories. This allowed assessing the degree of their similarity and identifying potentially dangerous deviations. Additionally, the application of the DTW algorithm to account for time shifts contributed to a more accurate analysis of changes in the ship's trajectory, increasing the effectiveness of trajectory classification by risk level.

At the second stage, trajectory clustering was implemented using the DBSCAN and D-KMEANS algorithms. DBSCAN was applied to identify clusters based on the density of points in space, considering the distance to the shoreline, while D-KMEANS refined cluster boundaries and improved classification accuracy. This enabled the isolation of dangerous zones based on spatiotemporal characteristics of ship movement.

At the third stage, the Douglas-Peucker algorithm was used for trajectory compression. This algorithm provided a reduction in the number of points in the trajectories without losing key characteristics, significantly improving the efficiency of processing large data volumes and preserving clustering accuracy.

The fourth stage involved assessing ship movement stability using Fourier analysis. The Fourier transform allowed detecting frequency components of ship movement oscillations, helping identify instability caused by changes in course or speed. Analyzing the frequency characteristics of the ship's movement allowed identifying oscillations that may indicate a lack of experience or navigator qualification.

At the fifth stage, fuzzy trajectory clustering was used with the Gaussian Mixture Model (GMM). The application of GMM allowed modeling the probabilities of points belonging to specific clusters, enabling the prediction of dangerous trajectories and identifying high-risk zones, considering the uncertainty of navigation parameters.

The final, sixth stage concerned trajectory prediction using a multilayer neural network (MLP). The MLP model predicted future ship coordinates based on current data such as speed, course, and position. The prediction proved effective and allowed timely correction of the ship's movement, ensuring safety under complex navigation conditions.

The proposed comprehensive method for predicting dangerous ship trajectories under uncertainty of navigator actions confirmed its effectiveness, providing ship trajectory prediction and dangerous zone detection at an accuracy level of 72–81 %, which allows improving navigation safety under complex conditions.

Conflict of interest

The authors declare that they have no conflict of interest in relation to this research, whether financial, personal,

authorship or otherwise, that could affect the research and its results presented in this paper.

Financing

The research was performed without financial support.

Data availability

The manuscript has no associated data.

Use of artificial intelligence

The authors confirm that they did not use artificial intelligence technologies when creating the presented work.

References

1. Nosov, P., Koretsky, O., Zinchenko, S., Prokopchuk, Y., Gritsuk, I., Sokol, I., Kyrychenko, K. (2023). Devising an approach to safety management of vessel control through the identification of navigator's state. *Eastern-European Journal of Enterprise Technologies, 4 (3 (124))*, 19–32. https://doi.org/10.15587/1729-4061.2023.286156
2. Ponomaryova, V., Nosov, P., Ben, A., Popovych, I., Prokopchuk, Y., Mamenko, P. et al. (2024). Devising an approach for the automated restoration of shipmaster's navigational qualification parameters under risk conditions. *Eastern-European Journal of Enterprise Technologies, 1 (3 (127))*, 6–26. https://doi.org/10.15587/1729-4061.2024.296955
3. Hrytsuk, I., Nosov, P., Ponomarova, V., Diahyleva, O. (2023). Reduction of navigation risks by using fuzzy logic to automate control processes under uncertainty. *Nauka i Tekhnika Sohodni, 6 (20)*, 8–22. https://doi.org/10.52058/2786-6025-2023-6(20)-8-22
4. Liang, M., Liu, W., Gao, R., Xiao, Z., Zhang, X., Wang, H. (2024). *A Survey of Distance-Based Vessel Trajectory Clustering: Data Pre-processing, Methodologies, Applications, and Experimental Evaluation*. https://doi.org/10.48550/arXiv.2407.11084
5. Ma, L., Shi, G., Li, W., Jiang, D. (2023). A Direction-Preserved Vessel Trajectory Compression Algorithm Based on Open Window. *Journal of Marine Science and Engineering, 11 (12)*, 2362. https://doi.org/10.3390/jmse11122362
6. Alam, M. M., Spadon, G., Etemad, M., Torgo, L., Milios, E. (2024). Enhancing short-term vessel trajectory prediction with clustering for heterogeneous and multi-modal movement patterns. *Ocean Engineering, 308*, 118303. https://doi.org/10.1016/j.oceaneng.2024.118303
7. Zhang, R., Chen, X., Ye, L., Yu, W., Zhang, B., Liu, J. (2024). Predicting Vessel Trajectories Using ASTGCN with StemGNN-Derived Correlation Matrix. *Applied Sciences, 14 (10)*, 4104. https://doi.org/10.3390/app14104104
8. Yang, X., Han, Z., Zhang, Y., Liu, H., Liu, S., Ai, W., Liu, J. (2024). VEPO-S2S: A VEssel Portrait Oriented Trajectory Prediction Model Based on S2S Framework. *Applied Sciences, 14 (14)*, 6344. https://doi.org/10.3390/app14146344
9. Luo, S., Zeng, W., Sun, B. (2023). Contrastive Learning for Graph-Based Vessel Trajectory Similarity Computation. *Journal of Marine Science and Engineering, 11 (9)*, 1840. https://doi.org/10.3390/jmse11091840
10. Zhu, Q., Xi, Y., Hu, S., Chen, Y. (2023). Exploring the behavior feature of complex trajectories of ships with Fourier transform processing: a case from fishing vessels. *Frontiers in Marine Science, 10*. https://doi.org/10.3389/fmars.2023.1271930
11. Li, Y., Yu, Q., Yang, Z. (2024). Vessel Trajectory Prediction for Enhanced Maritime Navigation Safety: A Novel Hybrid Methodology. *Journal of Marine Science and Engineering, 12 (8)*, 1351. https://doi.org/10.3390/jmse12081351
12. Zhang, X., Zhou, S. (2023). Vessel Trajectory Data Compression Algorithm considering Critical Region Identification. *Journal of Advanced Transportation, 2023*, 1–18. https://doi.org/10.1155/2023/8831371
13. Donandt, K., Söffker, D. (2024). *Incorporating Navigation Context into Inland Vessel Trajectory Prediction: A Gaussian Mixture Model and Transformer Approach*. https://doi.org/10.48550/arXiv.2406.02344
14. Yang, C.-H., Wu, C.-H., Shao, J.-C., Wang, Y.-C., Hsieh, C.-M. (2022). AIS-Based Intelligent Vessel Trajectory Prediction Using Bi-LSTM. *IEEE Access, 10*, 24302–24315. https://doi.org/10.1109/access.2022.3154812
15. Aguilar-Ibanez, C., Suarez-Castanon, M. S., García-Canseco, E., Rubio, J. de J., Barron-Fernandez, R., Martinez, J. C. (2024). Trajectory Tracking Control of an Autonomous Vessel in the Presence of Unknown Dynamics and Disturbances. *Mathematics, 12 (14)*, 2239. https://doi.org/10.3390/math12142239
16. Li, J. H., Kang, H., Kim, M. G., Lee, M. J., Cho, G. R. (2022). Asymptotic Trajectory Tracking of Underactuated Non-minimum Phase Marine Vessels. *IFAC-PapersOnLine, 55 (31)*, 281–286. https://doi.org/10.1016/j.ifacol.2022.10.443
17. Oka Widyantara, I. M., Noven Hartawan, I. P., Ngurah Eka Karyawati, A. A. I., Er, N. I., Artana, K. B. (2023). Automatic identification system-based trajectory clustering framework to identify vessel movement pattern. *IAES International Journal of Artificial Intelligence (IJ-AI), 12 (1)*, 1. https://doi.org/10.11591/ijai.v12.i1.pp1-11
18. Xu, Y., Zhang, J., Ren, Y., Zeng, Y., Yuan, J., Liu, Z., Wang, L., Ou, D. (2022). Improved Vessel Trajectory Prediction Model Based on Stacked-BiGRUs. *Security and Communication Networks, 2022*, 1–17. https://doi.org/10.1155/2022/8696558
19. Cahapin, E. L., Malabag, B. A., Santiago Jr., C. S., Reyes, J. L., Legaspi, G. S., Adrales, K. L. (2023). Clustering of students admission data using k-means, hierarchical, and DBSCAN algorithms. *Bulletin of Electrical Engineering and Informatics, 12 (6)*, 3647–3656. https://doi.org/10.11591/eei.v12i6.4849
20. Yang, L., Liu, J., Wang, Y. (2024). Application of adaptive douglas-peucker with acceleration algorithm in ship trajectory compression. *2024 IEEE Conference on Artificial Intelligence (CAI)*, 206–209. https://doi.org/10.1109/cai59869.2024.00045
21. Sun, B., Wu, X., Chen, X., Zou, Z., Li, Q., Ren, B. (2024). Parameter Estimation of Sub-/Super-synchronous Oscillation Based on Interpolated All-phase Fast Fourier Transform with Optimized Window Function. *Journal of Modern Power Systems and Clean Energy, 12 (4)*, 1031–1041. https://doi.org/10.35833/mpce.2023.000179
22. Asri, S. D. (2024). Underwater Image Segmentation with the GMM (Gaussian Mixture Model) Algorithm. *JSAI (Journal Scientific and Applied Informatics), 7 (2)*, 253–257. https://doi.org/10.36085/jsai.v7i2.6418
23. Sambasiva Rao, R. (2014). Mathematical Neural Network (MaNN) Models Part VI: Single-layer perceptron [SLP] and Multi-layer perceptron [MLP] Neural networks in ChEM- Lab. *Journal of Applicable Chemistry, 3*, 2209–2311.

Victoria Ponomaryova, *PhD Student, Department of Ship Electrical Equipment and Automatic Devices Operation, Kherson State Maritime Academy, Kherson, Ukraine, ORCID: https://orcid.org/0000-0001-9660-1772*

✉***Pavlo Nosov,*** *PhD, Associate Professor, Department of Ship Computer Systems and Networks, Kherson State Maritime Academy, Kherson, Ukraine, e-mail: pason@ukr.net, ORCID: https://orcid.org/0000-0002-5067-9766*

✉*Corresponding author*