UDC 004.58:004.8

METHOD OF DECISION SUPPORT FOR NAVIGATORS FOR AUTOMATED CONTROL OF VESSEL TRAFFIC SAFETY BASED ON ECDIS DATA

Ponomaryova Victoria, *Graduate student, Kherson State Maritime Academy, Ukraine, e-mail: vikkiivan@gmail.com; https://orcid.org/0000-0001-9660-1772.*

Objective of the research is to develop a method for integrating automated decision support tools for navigator on the bridge of a sea vessel, considering the factors of uncertainty in the completeness of ECDIS data.

The primary problem of the research addressed is the need for accurate and efficient Decision Support Systems (DSS) that account for uncertainties in electronic navigational data.

Research Methodology involves the development of automated modules: an OCR processing module for ECDIS images using the Tesseract library, a module for comparing textual and geolocation data from ECDIS screenshots using text comparison algorithms and geolocation calculations (Haversine formula), a geographic data visualization module on an interactive map using the Folium library, and a decision support module for navigators that includes analyzing of navigational data, determining their similarity, and providing recommendations.

Research results demonstrate that the developed DSS significantly enhances navigation safety, reduces travel time by 7% to 18%, and saves fuel, lubricants, and electricity. It increases the accuracy and efficiency of navigation by automating OCR processing (capturing ECDIS screenshots in real-time, preprocessing images to enhance OCR accuracy, extracting text, and saving it to files), text and geolocation data comparison (analyzing information and geographic data to determine their similarity, loading data from files, and calculating similarity), and data visualization on an interactive map (creating maps with markers and routes based on geographic data).

Practical significance of the research lies in improving navigational decision-making processes, reducing navigator workload, enhancing situational awareness, and minimizing collision risks in maritime navigation. The DSS automates critical aspects of navigation operations, reducing the likelihood of human errors.

Prospects for further research include improving data integration methods to enhance the accuracy and reliability of the DSS. Future work will benefit from the use of artificial neural networks to obtain better approximations. An important aspect of DSS development is identifying navigator qualification parameters to ensure logical conclusions regarding their actions and prevent undesirable consequences. Further research is necessary to expand and verify the effectiveness of the DSS in real maritime navigation conditions, which will improve algorithms for analyzing large volumes of data and integrating artificial intelligence to provide more adaptive and autonomous solutions.

Bibliography: 23, figures 11.

Key words: *ECDIS; automation; risk; uncertainty; automated control systems; intelligent systems; human factor; maritime safety; qualification parameters; identification.*

DOI: 10.33815/2313-4763.2024.1.28.022-040

Introduction. Keeping watch in adverse conditions increases the demands on navigators, their qualifications, experience, and speed of making rational decisions [1]. Considering that in complex situations where the full spectrum of navigation parameters is uncertain, the navigator has terminal limitations for forming a comprehensive real-time picture of the situation [2]. In some cases, enhancing safety can be assisted by autopilots, pilots, or tugs, but ultimately only the captain is responsible for the safety of navigation [3]. The reduction in the number of watch team members, combined with an increase in the number of navigational information systems, overloads navigators in the waters of seaports, canals, etc. [4]. Additionally, the intensity of traffic and the need to comply with COLREG rules compel captains to choose decision-making strategies very closely and carefully, which, in conditions of limited experience despite qualifications, can lead to non-standard and even critical situations [5]. All of this necessitates new informational approaches in the development of decision support technologies that narrow the navigator's focus on key aspects, processing a significant flow of information in real-time.

For the creation of decision support systems (DSS) for navigators, the application of big data and analytics for ship identification and collision risk analysis using deep reinforcement learning is considered, which allows for route optimization and enhances navigation safety. A separate direction is the use of artificial intelligence, automated systems, and fully autonomous robots to

improve port logistics and cargo transportation, promoting efficiency and reducing human intervention [6].

Thus, the development of specialized DSS in navigation systems will not only improve the accuracy of forecasts but also provide additional levels of control and support. Such systems will help analyze large volumes of data in real-time, assisting the captain in making more informed decisions while managing the ship's movement.

Problem Statement. Given the above, the development of a navigator decision support system (DSS) under conditions of partial uncertainty directs research into the following scientific directions:

1. Reducing the workload on navigators. With the increasing volume of information coming from navigation systems, navigators often face overload, which can lead to errors. It is important to develop systems that will effectively filter and prioritize information, displaying only the most critical data to the navigators.

2. Improving situational awareness. In complex navigational conditions, especially in narrow channels and ports, navigators may not have a complete picture of the situation. It is important to implement technologies that will help create a comprehensive and accurate picture of the navigational situation.

3. Minimizing collision risks. Despite the introduction of new technologies, the risk of collisions remains high, especially in conditions of intense traffic. Decision support systems should include algorithms for analyzing ship trajectories and warning about deviations from a safe course.

4. Optimizing ship movement. To increase navigation efficiency and reduce risks, it is necessary to use artificial neural network training to identify the level of danger regarding the ship's position and further optimize ship routes considering current conditions.

5. Analyzing navigator qualifications. It is important to develop computer programs to identify the qualification level of the navigator and analyze the impact of qualification levels on ship management methods in special and dangerous navigation areas.

Considering these scientific directions, a critical review of scientific sources will be conducted.

When considering a decision support system using radio communication for small vessel captains in the Caribbean Sea, the study noted the use of artificial neural networks. However, it lacks a detailed description of the model and comparison with other approaches, necessitating further research to confirm the results [7]. The integration of artificial intelligence, big data, and remote control for risk management in autonomous navigation is theoretically substantiated, but lacks sufficient methodology or data to confirm effectiveness, requiring practical testing of the model [8].

The method for determining collision risks using AIS data has limitations due to data transmission delays and incompleteness, indicating a need for empirical verification of its effectiveness in real navigation conditions [9]. To improve data transmission, the use of VDES instead of AIS is proposed. Although the new standard promises significant improvements, the study lacks real-world data to confirm the system's effectiveness in maritime navigation [10].

The integration of sensor technologies to improve the accuracy and reliability of navigation systems, as well as ensuring cybersecurity, is an important aspect of the research. Additionally, the need for improved personnel qualifications is emphasized [11]. The use of artificial intelligence, machine learning, big data analytics, and the Internet of Things for enhancing maritime navigation safety highlights the importance of processing large volumes of data and ensuring cybersecurity [6].

The analysis of methods for detecting marine objects using RGB cameras underscores the challenges associated with variable lighting conditions and weather phenomena. However, the application of deep learning methods requires significant computational resources [12].

The development of intelligent systems for predicting marine conditions using deep learning algorithms and image processing shows promise but needs real-world validation to confirm effectiveness [13]. The use of various sensors for data collection on the navigational environment

emphasizes the importance of visual perception and target detection at distances up to 6 nautical miles. Issues such as "blind spots" and signal loss are noted [14].

The integration of AIS data for modeling and forecasting the trajectories of other vessels using deep learning technologies underscores the need for additional data processing and verification [15]. In certain situations, the approach for Smart Maritime Autonomous Surface Vessel (SMASV) based on an improved Soft Actor-Critic (SAC) algorithm demonstrates high efficiency but requires real-world confirmation [16].

Probabilistic models also show potential, as exemplified by the dynamic Bayesian network (DBN) model for risk analysis in intelligent navigation of autonomous ships. This approach is effective but requires precise data and significant computational resources [17]. The application of the RMA model for classifying images of navigational signs achieves high accuracy but must consider the limitations of real navigation conditions and parameter settings [18]. Developing methods to ensure the functional resilience of navigation systems in non-standard situations is crucial, highlighting the need for further research to confirm the effectiveness of these methods [19].

The new AN-YOLOv4 algorithm for detecting navigational signs increases accuracy to 92% through the use of DCGAN and pyramid image methods, but real-world technical challenges must be addressed [20]. In other studies, the intelligent decision-making system for analyzing the state of the ship and ensuring hull balance uses IoT technologies for monitoring and data collection, but requires high-quality data and algorithms [21].

To account for uncertainty, fuzzy set theory and fuzzy logic are recommended for expert evaluations, emphasizing the need for real-world implementation of these methods [22]. However, such systems require a high level of data verification from experts. The use of virtual reality (VR) for crew training creates an interactive and realistic environment, but demands significant computational resources and user adaptation [23].

Thus, the development of a navigator DSS under conditions of partial uncertainty aims to reduce the workload on navigators, improve situational awareness, minimize collision risks, optimize ship movement, and analyze navigator qualifications. The main problem lies in the impact of a large amount of data and complex navigation conditions, which can lead to errors and dangers. To solve this problem, it is necessary to develop a system that effectively filters and provides the most critical information, improves awareness of the navigational situation, and takes into account the navigator's qualification level to enhance navigation safety and efficiency.

Research Purpose and Objectives. The aim of the research is to develop a method for integrating automated decision support tools for a navigator on the bridge of a sea vessel, considering the factors of uncertainty in the completeness of ECDIS data. To achieve this goal, a number of tasks must be solved to create a navigator DSS by developing automated modules:

1. Develop a module for automated OCR processing of images and text recognition on ECDIS display images in real time. The software module is designed to capture screenshots, preprocess images to enhance OCR accuracy, extract text from images using the Tesseract library, and save the extracted text to a file. Accomplishing this task will prepare the data for further analysis and ensure high text recognition accuracy.

2. Develop a module for comparing textual data and geolocations to analyze information and geographic data between different ECDIS screenshots to determine their similarity. The module includes loading data from files, analyzing key values using text comparison algorithms, and calculating the similarity of geolocations. Automating this process will speed up data analysis and support informed decision-making. Completing this task is essential for subsequent stages as it provides reliable data for comparison and analysis.

3. Develop a module for visualizing geographic data on a map, which includes creating interactive maps with markers and routes based on geographic data. The use of the Folium library is planned to create the map, add markers for each coordinate, visualize the ship's route and activity zone, and save the map in HTML web format. The data visualization module on the map will

ensure clarity and convenience for further analysis within the DSS, promoting a better understanding of movements and geographic patterns.

4. Develop a decision support module for a navigator, which will include comparing navigational data between pattern files, determining similarity between them, and providing recommendations based on this similarity. The module will load data from files, compare key values, calculate the similarity of textual data and geolocations, and output recommendations from an expert dictionary. Automating this process will effectively analyze navigational data, identify pattern files with high similarity, and provide specialized recommendations. Completing this task will form the core data and analytics for strategic decision-making by a navigator.

Primary Research Material. To achieve the research objective, it is necessary to develop integration modules for incorporating these DSS tools into the navigator's workflow on the bridge of a sea vessel. This involves addressing several key tasks: ensuring the completeness and accuracy of electronic navigational data (ECDIS), automating data processing and analysis, and providing navigators with timely and actionable recommendations.

To accomplish these tasks, the creation of automated modules, which constitute the core components of the navigator DSS, is proposed. These include modules for optical character recognition (OCR) to process ECDIS images, geolocation comparison to analyze and validate navigational data, and visualization tools to enhance situational awareness through interactive maps. Additionally, the decision support module is designed to compare navigational patterns and provide specialized recommendations based on the analysis of textual and geospatial data.

To create the navigator DSS, the route to the port of Lagos, Timkan was chosen (Figure 1). In collaboration with an expert, Captain of the long voyage, PhD, Pavlo Momenko, an action dictionary was compiled for each stage of the route, according to ECDIS data.

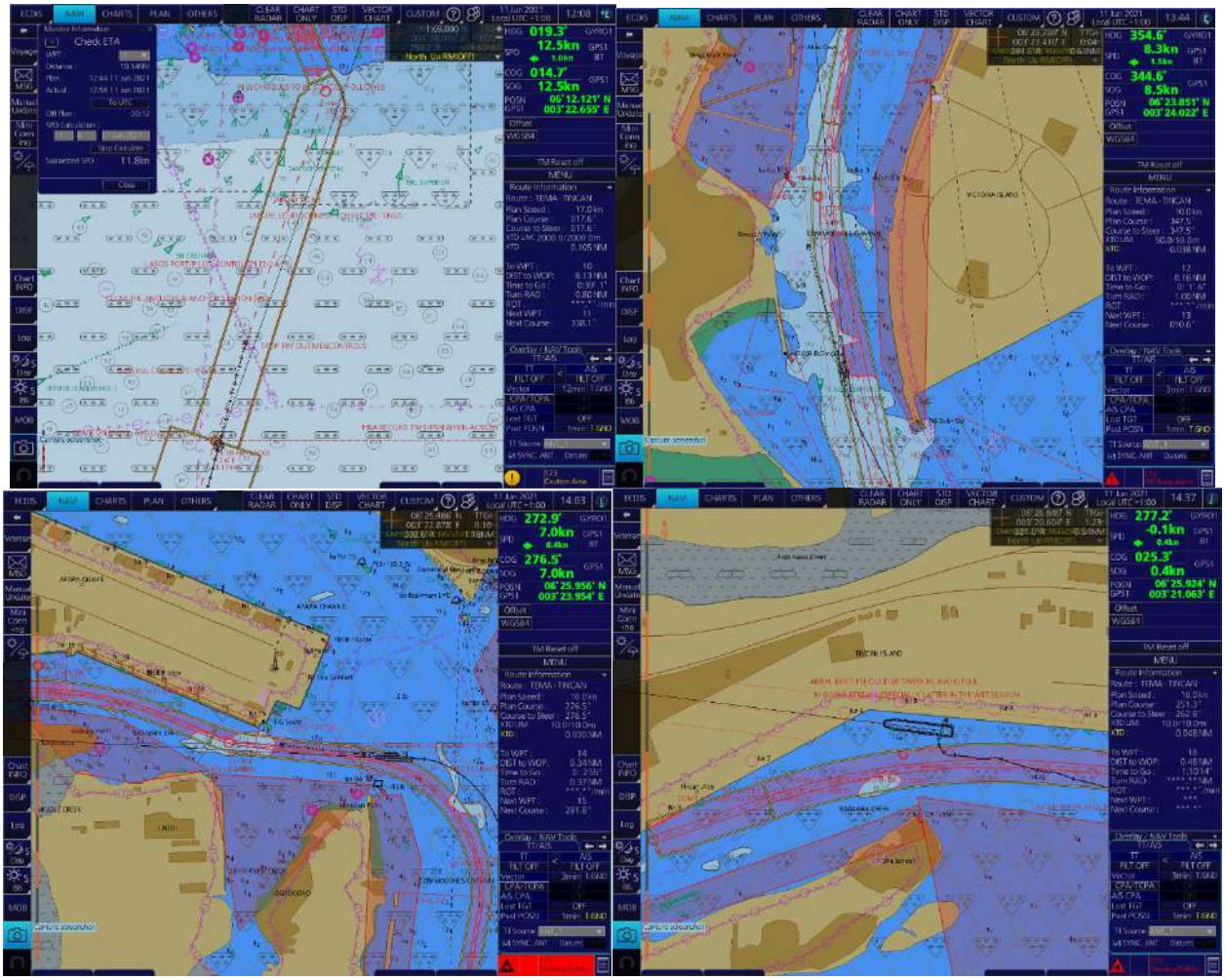


Figure 1 – Fragments-screenshots of the ECDIS TRANSAS system display in the port of Lagos

Considering the data from the ECDIS TRANSAS navigational information systems, the stages of the navigator's actions (expert system dictionary) were determined, which were subsequently planned to be integrated into the navigator DSS.

Initial data on the stages for creating the navigator DSS:

Stage 1. Initial Preparation: The captain and the bridge team review the route plans and port information, considering specific restrictions or requirements.

Stage 2. ECDIS №1 (Screenshot 20210611110841). Preparation of the Pilot Ladder: Setting up the ladder according to the pilot's instructions (side, height, straps). Speed adjustment: controlling the ship's speed according to the pilot's instructions, reducing speed if necessary. Visual and radar monitoring: monitoring the navigational situation using radars, with special attention to dredging vessels. Using the echo sounder: monitoring the water depth under the keel, cross-checking with the electronic chart. Fixing the ship's position: marking the position on the e-chart at 5-7 minute intervals. Ship movement: moving at a speed of 12.5 knots, adjusting the speed. Course angle 19.30, ground course 14.70.

Stage 3. ECDIS №2 (Screenshot 20210611115028), ECDIS №3 (Screenshot 20210611115557), ECDIS №4 (Screenshot 20210611120848), ECDIS №5 (Screenshot 20210611121957): Communication: maintaining contact with the port's vessel traffic service and the pilot via VHF channels 16/12. Ship speed adjustment: reducing speed to DSAH (dead slow ahead) to 4 knots, which is the minimum safe speed for ship maneuverability using the rudder. Navigational planning and control: more carefully monitoring the depth under the keel in accordance with the ship's draft, especially when approaching the 30-meter isobath, monitoring the course angle 004.5° and ground course 354.8°. Ship maneuvering: performing maneuvers with reversing the main engine and using the bow thruster, awaiting the pilot.

Stage 4. Pilot Boarding: Establishing radio communication: the captain establishes radio contact using the VHF receiver on channels 16/12. Monitoring the pilot boat: visually observing the approach of the pilot boat. Managing ship speed: moving the ship at a speed of 6 knots, as indicated by the pilot. Navigational planning: changing the ship's course towards the entrance channel, defined by navigation buoys. Checking the pilot ladder: the watch officer goes to the main deck to the pilot boarding area to check the pilot ladder according to international requirements (Resolution A.1045(27) and IMAP Notice № 849).

Stage 5. ECDIS №6 (Screenshot 20210611122409): Speed and course management of the ship: the ship moves at a speed of 6 knots, set by the pilot, the captain adjusts the course to the entrance channel according to the route in ECDIS. Coordination with the pilot boat: the watch officer monitors the pilot's boarding and informs the captain after the safe departure of the boat. Pilot reception: the watch officer accompanies the pilot to the bridge, the captain provides the pilot with information about the ship through the pilot card and the main engine characteristics and ship's maneuvering characteristics. Beginning of the second maneuvering stage: after providing information to the pilot, the second maneuvering stage begins – pilotage, which requires coordination between the captain, the watch officer, and the pilot for the ship's safety when approaching the port and maneuvering in the port area.

Stage 6. ECDIS №7 (Screenshot 20210611122937), ECDIS №8 (Screenshot 20210611124034): Maneuvering along the fairway: the ship follows the approach fairway, marked by red buoys on the left and green buoys on the right (IALA region A system). Controlling ship speed and course: ship speed (SOG) is 6.2 knots, course (COG) 348.8°. Using Parallel Indexing (PI): the captain uses PI to control the ship's position relative to the shoreline and landmarks, setting distance limits. Monitoring depth under the keel: tracking depths on the sea chart and cross-checking with echo sounder readings to ensure safe depth under the keel. Adjusting speed depending on conditions: speed increases to SAH (slow ahead) 8-10 knots for better maneuverability, and decreases to DSAH (dead slow ahead) 6-8 knots for precise maneuvering when approaching the dock.

Stage 7. ECDIS №9 (Screenshot 20210611124452), ECDIS №10 (Screenshot 20210611124902), ECDIS №11 (Screenshot 20210611125313), ECDIS №12 (Screenshot

20210611125928): Maneuver preparation: the bridge team carefully studies and plans maneuvers in difficult sections of the fairway, highlighted in the Harbour Approach & Manoeuvring Plan. Maneuvering during approach: performing a 90° turn to the left, changing the course to the required direction, then maneuvering to the right to 220°. Controlling speed and course: increasing speed to 8-10 knots for maneuverability, then reducing to 6-8 knots for precise maneuvering. Ensuring safety: strictly following the pilot's commands and helm adjustments with high attention. Using Parallel Indexing (PI): setting distance limits relative to landmarks or the shoreline for safe distances. Integration with ECDIS: information about the ship's speed, helm angle, and point of helm adjustment is added to the electronic chart. Responding to conditions: the captain remains attentive and ready to intervene in case of unsafe maneuver execution by the pilot.

Stage 8. ECDIS №13 (Screenshot 20210611130013), ECDIS №14 (Screenshot 20210611130326), ECDIS №15 (Screenshot 20210611130626): Coordination with the pilot and helmsman for maneuvers: close cooperation with the pilot for precise maneuvering, especially during 90° and 220° turns, evaluating performed maneuvers and readiness to intervene in case of accident risk. Adjusting ship speed: ship speed (SOG) varies between 7.4 and 7.2 knots during maneuvering. Navigational control using Parallel Indexing (PI): using PI to establish safe distances from the shoreline and moored vessels. Monitoring the suction effect: maintaining speed modes to minimize the risk of creating a suction effect that can damage the mooring lines of moored vessels.

Stage 9. ECDIS №16 (Screenshot 20210611130909), ECDIS №17 (Screenshot 20210611131354), ECDIS №18 (Screenshot 20210611131820): Navigation: the ship moves in port waterways at a speed of 7.6 knots (SOG), the ship's course (COG) changes from 250 to 350 degrees, indicating direction changes during maneuvering. Maneuvering operations: the captain coordinates maneuvers for safe passage near moored vessels, maintaining distance from the shoreline and other obstacles. Speed control: the captain controls the speed to minimize the risk of damaging moored vessels and their mooring lines due to the suction effect. Communication with the bridge: the captain maintains constant communication with the bridge and helmsman for quick and accurate execution of maneuvering commands.

Stage 10. ECDIS №19 (Screenshot 20210611132652): Preparation for towing: the ship reduces speed to 4-5 knots for safe approach to the tugs. The captain organizes mooring teams under the direction of the third mate on the bow and the second mate on the stern. Communication with tugs: commands for tugs are given through the pilot and the captain; direct communication between crews is excluded. Attention to the human factor: the captain focuses on avoiding errors due to language barriers or professional differences. Navigational parameters: the ship's course (COG) is 250°-278°, speed (SOG) 4-7 knots, indicating preparation for docking or towing.

Stage 11. ECDIS №20 (Screenshot 20210611132715): Controlling the ship's speed and direction: the ship's speed is reduced to 4.3 knots for controlled approach to the dock with the possibility of reversing. Cooperation with tugs: the captain coordinates actions with tugs after confirming their readiness. Reversing the main engine: depending on the distance to the dock and the ship's speed, the captain chooses between reversing the main engine or additional tug thrust. Ship handling: the rudder is used for precise control of the ship when approaching the dock, even at low speed.

Stage 12. ECDIS №21 (Screenshot 20210611133405), ECDIS №22 (Screenshot 20210611133627), ECDIS №23 (Screenshot 20210611133740), ECDIS №24 (Screenshot 20210611133903): Parallel docking of the ship: the captain coordinates actions to position the ship parallel to the dock, using tugs at a speed of 0.5 knots, then 0.0 knots. Main engine readiness for maneuver: the main engine is ready for any necessary maneuver at the captain's command. Preparation for an emergency: the boatswain and the mooring team are ready to use the anchor in case of an unforeseen situation. Mooring operations: the captain instructs the mooring teams on the bow and stern regarding the delivery of mooring lines to the dock. Completion of maneuvering: after securing the mooring lines and releasing the tugs, the captain completes the maneuvering, gives the command to stop using the main engine, and the pilot leaves the ship.

The data within the expert system dictionary allowed the transition to creating software modules for the navigator DSS as a whole.

1. Development of an automated OCR module for image processing and text recognition on ECDIS display images in real time on the bridge (Figure 2).

Detailed description of the program processes for OCR image processing with specified regions of interest (ROI):

Initialization and loading libraries:

Loading necessary modules: `import cv2, pytesseract, os, pyautogui, pygetwindow as gw, time.`

Setting configuration for pytesseract: `pytesseract.pytesseract.tesseract_cmd = r'C:\Program Files\Tesseract-OCR\tesseract.exe'.`

Capturing screenshots (take_screenshots):

Finding the window by title: `window = gw.getWindowsWithTitle(window_title)[0].`

Checking window activity and activating if necessary: `if not window.isActive: window.activate().`

Capturing and saving the screenshot: `screenshot = pyautogui.screenshot()` та `screenshot.save(filename),` де `filename = os.path.join(save_folder, f"{time.strftime ('%Y%m%d-%H%M%S')}.png").`

Repeating the process at a specified interval: `time.sleep(interval).`

Preprocessing the image for OCR (preprocess_image):

Reading the image: `image = cv2.imread(image_path).`

Converting the image to grayscale: `gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY).`

Extracting text from the image (extract_text_from_image):

Configuring and running OCR: `text = pytesseract.image_to_string(image, config='--oem 3 --psm 11').`

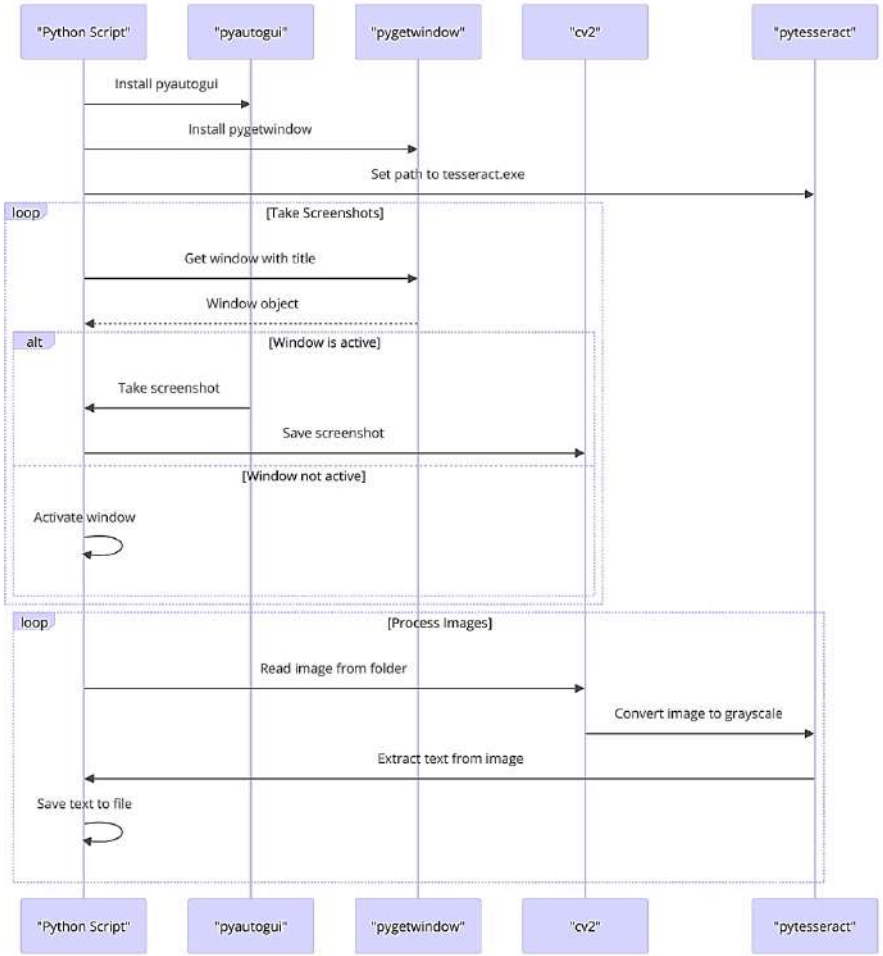


Figure 2 – Structure of the ECDIS Image Processing and Text Recognition Module

Saving text to a file (save_text_to_file):

Opening the file and writing the text: `with open(file_path, 'w', encoding='utf-8') as file: file.write(text).`

Processing all images in a folder (process_images_from_folder):

Iterating over files in the folder: `for root, dirs, files in os.walk(folder_path).`

Processing each image and saving the text: `preprocessed_image = preprocess_image(file_path), text = extract_text_from_image(preprocessed_image), save_text_to_file(text, text_file_path).`

Managing threads and user interaction:

Creating a thread: `screenshot_thread = Thread(target=take_screenshots, args=(window_title, screenshot_folder, 10)).`

Starting the thread: `screenshot_thread.start().`

Waiting for user input: `input("Натисніть Enter, коли закінчите працювати з програмою...").`

This approach ensures systematic and automated extraction of textual information from specific areas of the image, which can be useful for analyzing images with consistent structures, such as forms, maps, or other documents.

1.2. OCR image processing with specified regions of interest (ROI). Detailed ROI collection involves gathering key navigational data from ECDIS (Figure 3).

Initialization and loading libraries:

Importing libraries: `import cv2, pytesseract, os.`

Setting configuration for pytesseract: `pytesseract.pytesseract.tesseract_cmd = r'C:\Program Files\Tesseract-OCR\tesseract.exe'.`

Defining the region of interest (ROI) on the image (get_roi):

Getting a part of the image by specified coordinates: `return image[y1:y2, x1:x2].`

Performing OCR for the defined region (ocr_zone):

Defining ROI using get_roi: `roi = get_roi(image, x1, y1, x2, y2).`

Using pytesseract to recognize text in the area: `text = pytesseract.image_to_string(roi, config='--psm 7').`

Cleaning and returning the obtained text: `return text.strip().`

Image processing procedure (process_image):

Reading the image: `image = cv2.imread(image_path).`

Converting the image to grayscale to enhance OCR accuracy: `gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY).`

Defining zone coordinates and collecting data using OCR for each zone: `data = {key: ocr_zone(gray_image, coords) for key, coords in zones.items()}.`

Saving data to a file (save_data_to_file):

Opening the file for writing: `with open(file_path, 'w', encoding='utf-8') as file.`

Writing data to the file: file.write(f"{key}: `{value}\n" for key, value in data.items()).`

Main function for processing images in a folder (main):

Iterating over all images in the folder: `for file_name in os.listdir(folder_path).`

Checking the file format (accepts image files): `if file_name.lower().endswith(('.png', '.jpg', '.jpeg')).`

Processing each image and saving the results: `data = process_image(file_path), save_data_to_file(data, text_file_path).`

Outputting results or errors: `print(f"Дані з {file_name} збережено в {text_file_path}"), print(f"Помилка при обробці файлу {file_name}: {e}").`

This approach ensures systematic and automated extraction of textual information from specific areas of the image, which can be useful for analyzing images with consistent structures, such as forms, maps, or other documents.

For example, ECDIS №3 (Screenshot 20210611115557)_data, processed by the developed software module, has the following automatic identification: date: 11 Jun 2021; time: 12:55; heading_hdg: 000.5"; speed: 2.5kn; course_cog: 357.7; latitude: 06° 19.513' N; longitude: 003° 25.073" E; waypoint_wpt: To WPT: 10; distance_to_wpt: DIST to WOP: 0.32 NM; time_to_go: Time to Go: 0: 7'37"; next_wpt: ‘Next WPT: 1; next_course: ‘Next Course: 338.1";

distance_between_wpt: Distance: 2.43NM; plan_speed: Plan Speed: 17.0kn; plan_course: Plan Course: 017.6°.

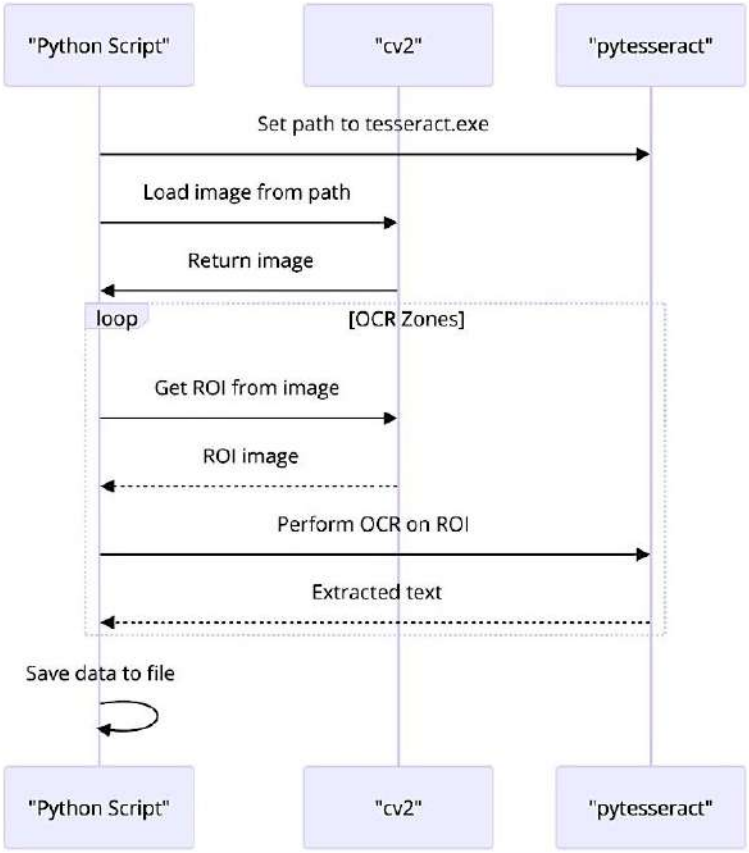


Figure 3 – Structure of the Image Processing Module with Navigational Data Zones from ECDIS

2. Development of a module for comparing textual data and geolocations to analyze information and geographic data between different ECDIS screenshots (Figure 4).

Initialization and loading libraries:

Importing libraries: `import os, from difflib import SequenceMatcher, from math import radians, cos, sin, asin, sqrt.`

Loading data from a file (load_data_from_file):

Opening the file and reading lines: `with open(file_path, 'r', encoding='utf-8') as file: lines = file.readlines().`

Extracting keys and values from file lines: `data = {line.split(':')[0].strip(): line.split(':')[1].strip() for line in lines if line.strip()}.`

Comparing data (compare_data):

Determining common keys for comparison, excluding 'date' and 'time': `keys_to_compare = set(test_data.keys()) & set(reference_data.keys()) - {'date', 'time'}.`

Calculating similarity for each key: `scores = [SequenceMatcher(None, test_data[key], reference_data[key]).ratio() for key in keys_to_compare].`

Returning the average similarity as a percentage: `return sum(scores) / len(scores) * 100 if scores else 0.`

Converting DMS to decimal coordinates (dms_to_decimal):

Cleaning the coordinate string and splitting it into parts: `parts = dms_str.replace('°', '').replace("'", '').replace('"', '').split().`

Converting degrees and minutes to decimal degrees, accounting for direction: `decimal = degrees + minutes / 60, if direction in ['S', 'W']: decimal = -decimal.`

Calculating the distance between two points (haversine):

Converting coordinates to radians and calculating the difference: `dlon = lon2 - lon1, dlat = lat2 - lat1.`

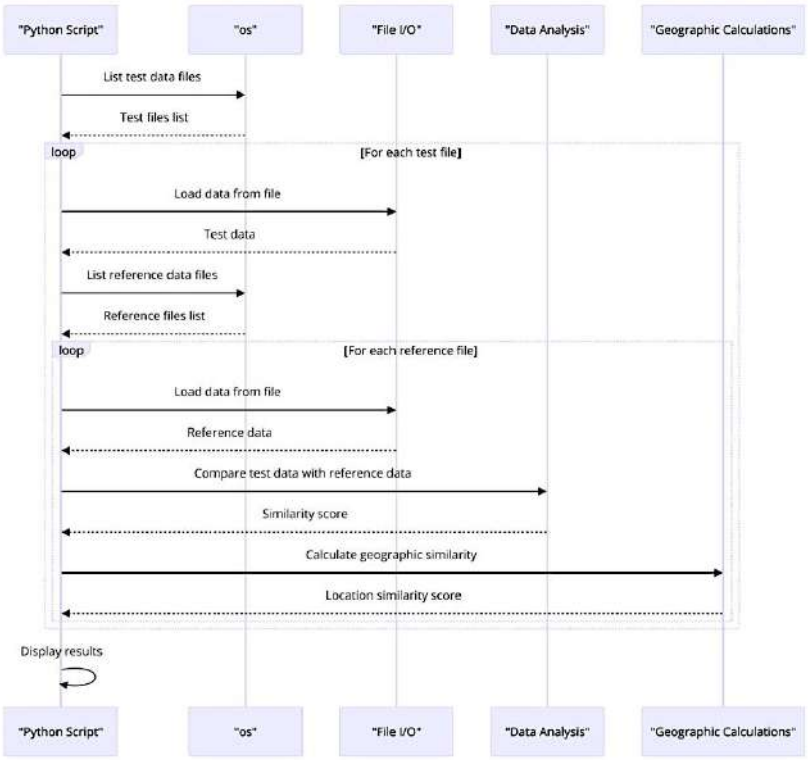


Figure 4 – Structure of the ECDIS Information and Geographic Data Analysis Module

Calculating the distance using the Haversine formula: `a = sin(dlat/2)**2 + cos(lat1) * cos(lat2) * sin(dlon/2)**2, c = 2 * asin(sqrt(a)), return c * r.`

Calculating location similarity (calculate_location_similarity):

Determining coordinates and calculating the distance: `distance = haversine(lat1, lon1, lat2, lon2).`

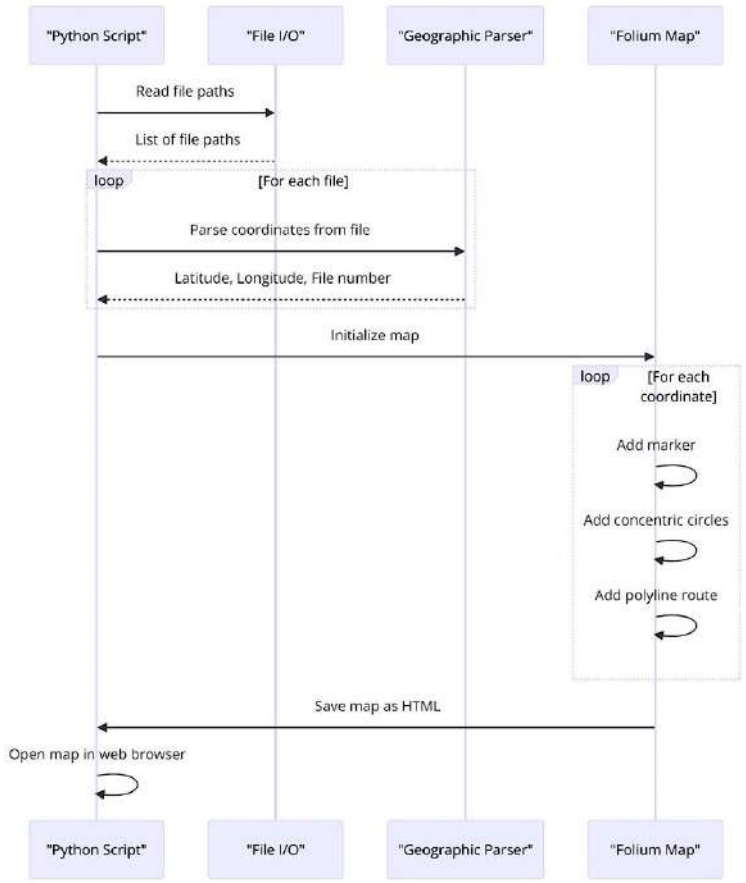


Figure 5 – Structure of the Geographic Data Visualization Module

Converting distance to percentage similarity: `return max(0, 100 - (distance / 1 * 100)).`

Main function for comparing files (main):

Loading and comparing data from test and reference files: `test_data = load_data_from_file(test_data_file), reference_data = load_data_from_file (reference_file_path), similarity = compare_data(test_data, reference_data).`

Outputting similarity results and sorting in descending order: `comparison_results.sort(key=lambda x: x[1], reverse=True).`

Checking similarity by geolocation: `for file_name, _, reference_data in comparison_results, loc_similarity = calculate_location_similarity(test_data, reference_data).`

This approach describes in detail how your program processes data, compares it based on textual information and geolocation, and organizes the output of results, allowing easy identification of files with a high level of similarity.

3. Development of a geographic data visualization module, which includes creating interactive maps with markers and routes based on geographic data.

Initialization and loading libraries:

Importing libraries: `import folium, import os, import glob, import webbrowser.`

Converting coordinates to decimal format (convert_to_decimal):

Converting degrees and minutes to decimal coordinates: `value = float(degrees) + float(minutes) / 60.`

Adjusting value for southern and western coordinates: `if direction in ['S', 'W']: value = -value.`

Parsing coordinates from files (parse_coordinates):

Opening the file and reading data: `with open(filename, 'r', encoding='utf-8') as file: lines = file.readlines().`

Selecting lines with latitude and longitude information: `latitude_line = next((line for line in lines if "latitude:" in line), None).`

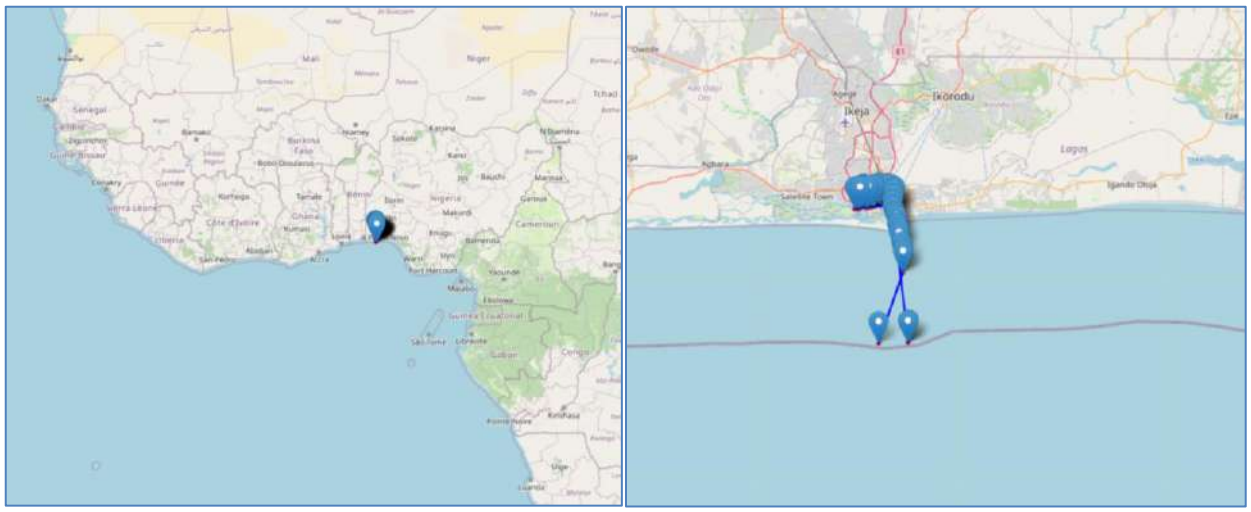
Cleaning latitude and longitude information and splitting into parts: `latitude_info = latitude_line.split('latitude:')[1]. strip().replace('°', ' ').replace("'", '').replace('"', '').`

Calculating decimal coordinates: `latitude = convert_to_decimal(lat_deg, lat_min, lat_dir).`

Creating the map and adding markers (Figure 6):

Initializing the map: `map = folium.Map(location=[0, 0], zoom_start=5).`

Adding markers to the map with popup text: `folium.Marker([lat, lon], tooltip=popup_text).add_to(map).`



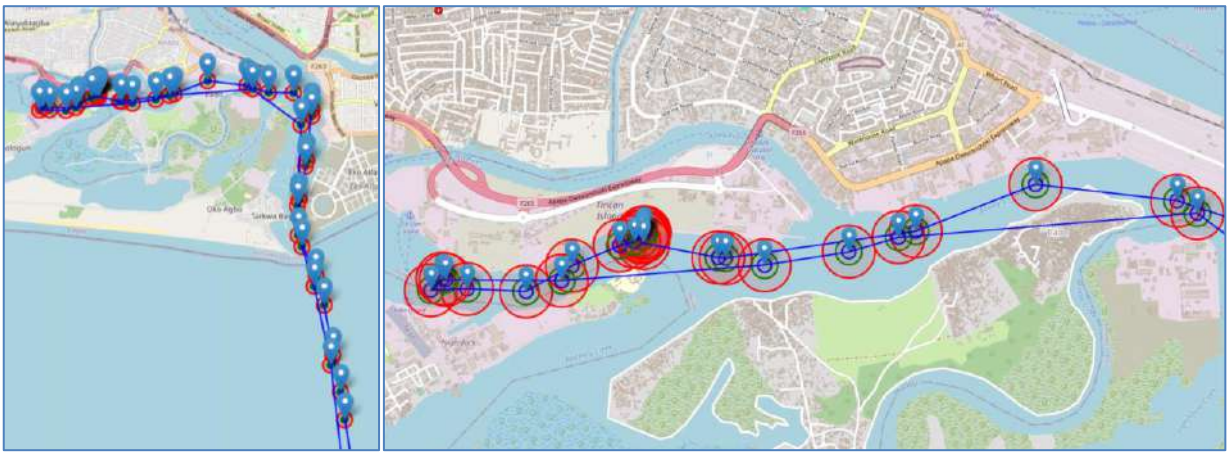


Figure 6 – Creating Interactive Maps with Route and Danger Zone Markings

Adding concentric circles and routes:

Adding different radii for skin coordinates: `folium.Circle(location=[lat, lon], radius=50, color='blue', fill=True).add_to(map).`

Creating a polyline to visualize the route: `folium.PolyLine(route_coordinates, color="blue", weight=2.5, opacity=1).add_to(map).`

Saving the map and opening it in a browser:

Saving the map to an HTML file: `map.save('map.html').`

Automatically opening the map in a web browser: `webbrowser.open('map.html', new=2).`

This program automates the process of creating an interactive map with markers that display locations from data in text files. It can visualize routes and activity zones, making it ideal for geographic analysis or tracking movements within research projects or logistics operations.

3.1. Detailed description of the program processes that read geographic positions from files and visualize them on an interactive map by adding markers indicating deviations from the base ship coordinate (Figure 7):

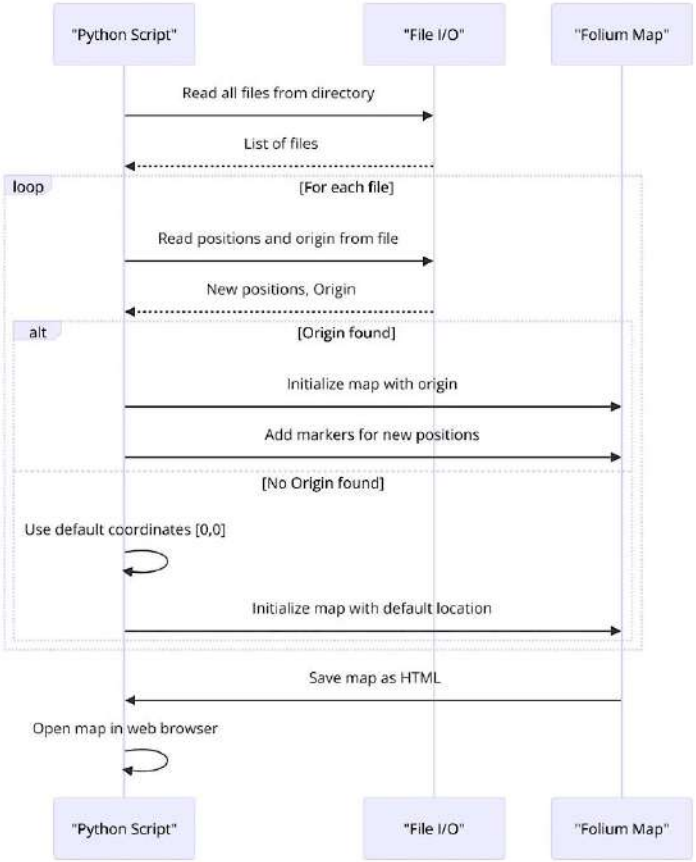


Figure 7 – Creating Interactive Maps with Markers Indicating Deviations from the Standard

Initialization and loading libraries:

Importing necessary libraries: `import folium, import webbrowser, import os.`

Reading positions and the initial point from a file (read_positions_and_origin):

Opening the file and reading all lines: `with open(file_path, 'r', encoding='utf-8') as file: lines = file.readlines().`

Finding the line containing the initial point label and parsing coordinates: `origin_coords = origin_parts[1].strip().strip('[]').split(','), origin_lat, origin_lon = [float(coord) for coord in origin_coords].`

Recording new positions after finding the initial point: if 'm' in line and origin_found: new_positions.append((lat, lon)).

Creating a map with all positions for the directory (create_map_with_all_positions):

Getting a list of files with specific suffixes: `all_files = [f for f in os.listdir(directory) if f.endswith("_s.txt")].`

Checking for the presence of files and selecting the first file to initialize the map: `first_file_path = os.path.join(directory, all_files[0]).`

Initializing the map with the first initial point: `my_map = folium.Map(location=first_origin, zoom_start=15).`

Adding markers to the map:

Reading positions and initial points from all files in the directory: `new_positions, origin = read_positions_and_origin(file_path).`

Adding markers for initial points and new positions: `folium.Marker(origin, icon=folium.Icon(color='red')).add_to(my_map)` та `for lat, lon in new_positions: folium.Marker([lat, lon]).add_to(my_map).`

Saving the map and opening it in a web browser:

Saving the map to an HTML file: `my_map.save('all_positions_map.html').`

Automatically opening the map in a web browser (Figure 8): `webbrowser.open ('all_positions_map.html', new=2).`

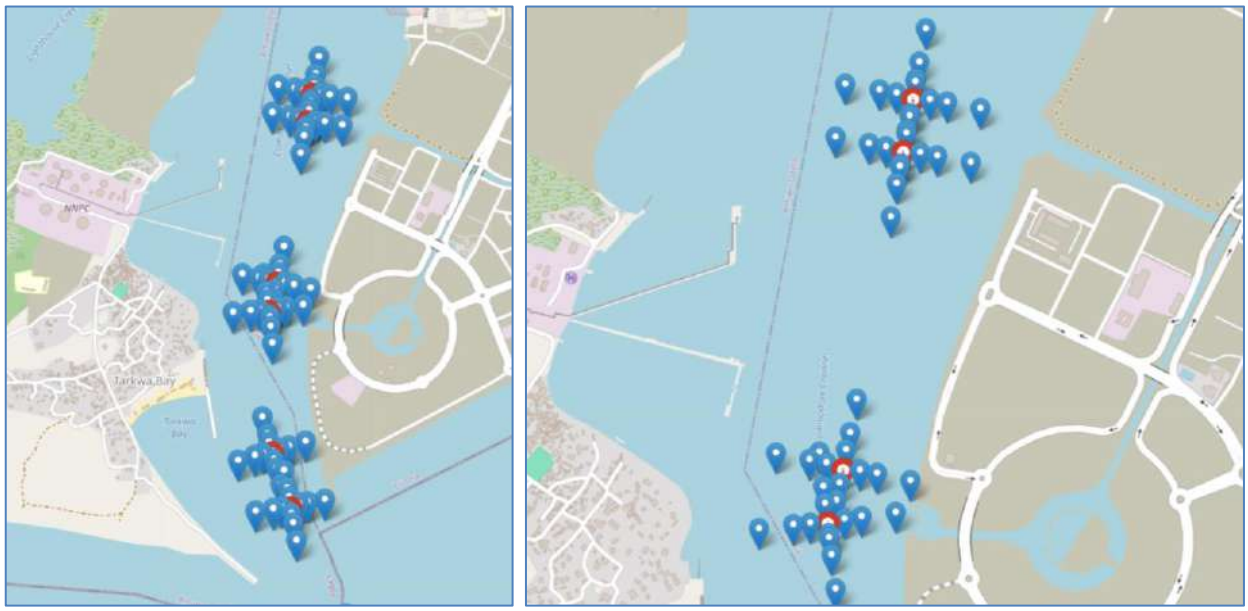


Figure 8 – Visualizing Multiple Markers on an Interactive Map by Safety Level

This program allows the visualization of geographic data from multiple files, integrating them into a single map, making it easy to analyze and compare geographic tracks and identify different geolocations specified in the data.

4. Development of a Decision Support Module for the Navigator, which will include comparing navigational data between pattern files, determining their similarity, and providing recommendations based on this similarity (Figure 9).

Initialization and loading libraries:

Importing necessary libraries: `import os, from difflib import SequenceMatcher, from math import radians, cos, sin, asin, sqrt.`

Loading data from a file (load_data_from_file):

Opening the file and reading lines: `with open(file_path, 'r', encoding='utf-8') as file: lines = file.readlines()`.

Extracting keys and values from file lines: `data = {line.split(':')[0].strip(): line.split(':')[1].strip() for line in lines if line.strip()}`.

Comparing data (compare_data):

Determining common keys for comparison, excluding 'date' and 'time': `keys_to_compare = set(test_data.keys()) & set(reference_data.keys()) - {'date', 'time'}`.

Calculating similarity for each key: `scores = [SequenceMatcher(None, test_data[key], reference_data[key]).ratio() for key in keys_to_compare]`.

Returning the average similarity as a percentage: `return sum(scores) / len(scores) * 100 if scores else 0`.

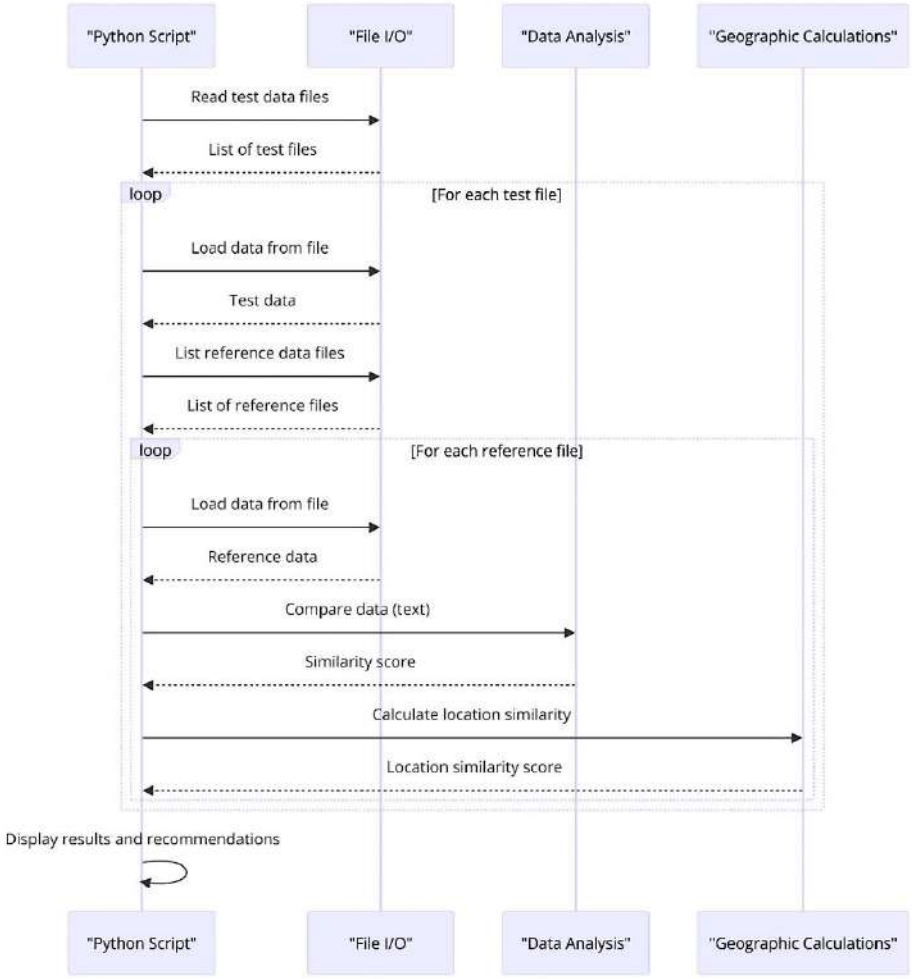


Figure 9 – Visualization of Multiple Markers on an Interactive Map by Safety Level

Converting DMS to decimal coordinates (dms_to_decimal):

Cleaning the coordinate string and splitting it into parts: `parts = dms_str.replace('°', '').replace("'", '').replace('"', '').split()`.

Converting degrees and minutes to decimal degrees, accounting for direction: `decimal = degrees + minutes / 60, if direction in ['S', 'W']: decimal = -decimal`.

Calculating the distance between two points (haversine):

Converting coordinates to radians and calculating the difference: `dlon = lon2 - lon1, dlat = lat2 - lat1`.

Calculating the distance using the Haversine formula: `a = sin(dlat/2)**2 + cos(lat1) * cos(lat2) * sin(dlon/2)**2, c = 2 * asin(sqrt(a)), return c * r`.

Calculating location similarity (calculate_location_similarity):

Determining coordinates and calculating the distance: `distance = haversine(lat1, lon1, lat2, lon2)`.

Converting distance to percentage similarity: `return max(0, 100 - (distance / 1 * 100))`.

Main function for comparing files (main):

Loading and comparing data from test and reference files: `test_data = load_data_from_file(test_data_file), reference_data = load_data_from_file (reference_file_path), similarity = compare_data(test_data, reference_data).`

Outputting similarity results and sorting in descending order: `comparison_results.sort(key=lambda x: x[1], reverse=True).`

Checking similarity by geolocation: `for file_name, _, reference_data in comparison_results, loc_similarity = calculate_location_similarity(test_data, reference_data).`

Providing recommendations based on comparison results:

Finding files with the highest similarity rating and outputting corresponding recommendations from the dictionary: `if top_file in recommendations: for rec in recommendations[top_file]: print(rec).`

This program automates the comparison of navigational data by determining similarity across various parameters, identifying the relevant file(s) to provide specialized recommendations based on this data (Figure 10).

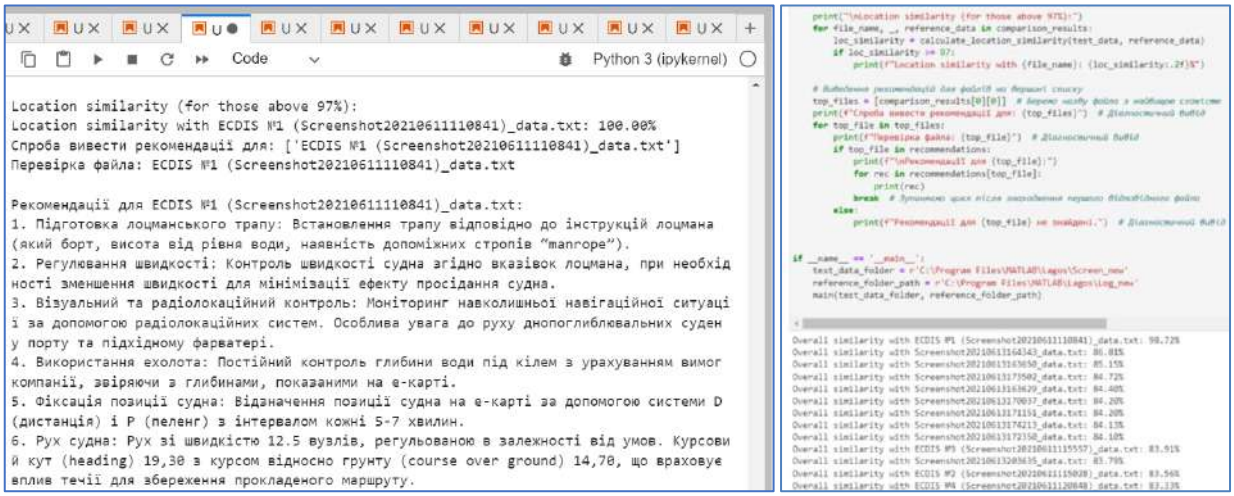


Figure 10 – Implementation of the Navigator Decision Support System

4.1. The functionality of loading point coordinates from files, comparing them with a reference point, calculating the distance between them, and providing recommendations based on point identifiers is presented in the following algorithm.

Loading coordinates: The function load_coordinates_from_file(filepath) reads the coordinates of points from a file, where the first line contains the reference point, and the subsequent lines contain the identifiers and coordinates of other points. The coordinates are stored in a dictionary with identifiers as keys.

Analyzing points: The function analyze_points(reference_point, points) calculates the Euclidean distances between the reference point and other points, storing the results in a dictionary. The identifier of the point with the shortest distance to the reference point is determined.

Providing recommendations: The function get_advice(identifier) provides recommendations based on the direction encoded in the point identifier. For example, the direction "270°" indicates the need to adjust the course to the right.

Haversine formula: The function haversine(coord1, coord2) calculates the distance between two coordinates on the surface of a sphere (in meters), using the Earth's radius and the coordinates of the points in radians.

Finding the best match: The function find_best_match(reference_file_path, comparison_files_directory) compares the reference point with other points from files in the specified directory. First, the reference point is loaded from the reference file. Then, for each file in the directory, the coordinates are loaded, and the distance between the reference points is calculated using the Haversine formula. The file with the shortest distance to the reference point is determined, and recommendations are provided based on the identifier of the closest point (Figure 11).

```
Best matching file: C:/Program Files/MATLAB/Lagos/Neuro_learning\5_s.txt
Distance to the closest point: 100.07553549486686 meters
Best identifier: 100m 180В°
Recommendation: 100m Продовжуйте рух прямо із пришвидшенням.
```

Figure 11 – Implementation of the Navigator Decision Support System (Recommendations for Ship Maneuvering)

This process allows automated finding and comparing of point coordinates, assessing their proximity to the reference point, and obtaining directional movement recommendations.

Experiments conducted using the TRANSAS Wärtsilä Navi-Sailor ECDIS navigation simulators significantly enhanced the support of the navigator's actions. This was particularly noticeable during training as part of the "Navigation and Piloting" course, ECDIS module.

Overall, the developed Navigator Decision Support System, exemplified by the Lagos location, Port Kimkan, reduced the ship's travel time by 7% to 18% when applied. Besides ensuring navigation safety, the Navigator Decision Support System shortens the ship's route and consequently saves fuel and lubricants as well as electricity on the ship.

Conclusion. The research focused on developing an automated Decision Support System (DSS) for navigators, specifically targeting the enhancement of maritime safety and efficiency under conditions of partial uncertainty in ECDIS data. The DSS developed integrates various automated modules, each addressing a specific aspect of navigational decision-making, from OCR processing of ECDIS images to the comparison of geolocation data and visualization of geographic information. The following conclusions summarize the key findings and results of the research:

1. The developed DSS significantly reduces the workload on navigators by effectively filtering and prioritizing navigational information. This reduces the risk of information overload, which can lead to errors and critical situations. The OCR processing module automates the extraction of textual data from ECDIS images, ensuring that only the most relevant information is presented to the navigator.

2. By integrating geographic data visualization and comparison modules, the DSS improves situational awareness. Navigators can better understand and interpret the navigational environment through interactive maps with markers and routes, enhancing their ability to make informed decisions in real-time.

3. The DSS includes advanced algorithms for analyzing ship trajectories and warning of potential deviations from safe courses. This proactive approach to collision risk management helps in maintaining a safe distance from other vessels and obstacles, thereby minimizing the risk of collisions.

4. The use of artificial neural networks and data analysis techniques allows for the optimization of ship routes. The DSS identifies potential hazards and optimizes the ship's movement to avoid these hazards, thus improving overall navigation efficiency. Practical tests have shown that the DSS can reduce travel time by 7% to 18%, leading to significant savings in fuel, lubricants, and electricity.

5. The decision support module provides tailored recommendations based on the comparison of navigational data. By identifying patterns and similarities in data, the DSS offers strategic advice to navigators, which is crucial for effective decision-making, especially in complex and high-risk navigational scenarios.

6. Experiments conducted using TRANSAS Wärtsilä Navi-Sailor ECDIS navigation simulators confirmed the effectiveness of the DSS in enhancing navigational safety and efficiency. The practical application of the DSS during the "Navigation and Piloting" course demonstrated its capability to reduce the ship's travel time and improve overall navigation performance.

Practical significance. The research provides a robust and practical solution for enhancing maritime safety and efficiency through the development of an automated DSS. The system's ability to filter and prioritize information, improve situational awareness, and provide strategic recommendations significantly enhances the navigator's decision-making capabilities. The DSS not

only ensures safer navigation but also contributes to operational efficiency by reducing fuel consumption and travel time.

Prospects for further research. Prospects for further research involve improving data integration methods to enhance the accuracy and reliability of the navigator DSS. Future work will benefit from the use of artificial neural networks to obtain intermediate assessments in the form of approximations. Additionally, an important aspect of developing the navigator DSS is identifying the navigator's qualification parameters to ensure logical conclusions regarding their actions and prevent undesirable consequences. Further research is necessary both to expand and verify the effectiveness of the DSS in real maritime navigation conditions. This will allow the improvement of algorithms for analyzing large volumes of data and integrating artificial intelligence to provide more adaptive and autonomous solutions.

REFERENCES

1. Ponomaryova, V., Nosov, P., Ben, A., Popovych, I., Prokopchuk, Y., Mamenko, P., Dudchenko, S., Appazov, E., & Sokol, I. (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.

2. 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 1of Enterprise Technologies, 4 (3 (124)), 19–32. https://doi.org/10.15587/1729-4061.2023.286156.

3. Zinchenko, S., Kobets, V., Tovstokoryi, O., Nosov, P., & Popovych, I. (2023). Intelligent System Control of the Vessel Executive Devices Redundant Structure. In CEUR Workshop Proceedings (Vol. 3403, Paper 44, pp. 582-594). CEUR-WS.org.

4. Gritsuk I. V., Nosov P. S., Ponomaryova V. P., Diahyleva O. S. (2023). Reduction of navigation risks by using fuzzy logic to automate control processes under uncertainty. «Наука і техніка сьогодні» (Серія «Техніка»)»: журнал. № 6(20). С. 8–22.

5. Victoria Ponomaryova, Pavlo Nosov. (2023). Method of automated identi-fication of qualification parameters for marine operators under risk conditions // Науковий вісник Херсонської державної морської академії (Автоматизація та комп'ютерно-інтегровані технології): науковий журнал. – Херсон: Херсонська державна морська академія, № 26–27. С. 144–165.

6. Zhang, Mingyang & Zhang, Xinyu & Fu, Shanshan & Dai, Lei & Yu, Qing. (2024). Recent Developments and Knowledge in Intelligent and Safe Marine Navigation. MDPI. 219 pp. ISBN: 978-3-03928-624-9.

7. Banaszek, Andrzej & Lisaj, Andrzej. (2023). The Radiocommunication Support Decision System to Use in Distress Situations for Captains of Small Non-conventional Vessels Operating in the Caribbean Sea Area. Procedia Computer Science. 225. 765–774. https://doi.org/10.1016/j.procs.2023.10.063.

8. Luo, Jianan & Geng, Xiongfei & Li, Yabin & Yu, Qiaochan. (2022). Study on the Risk Model of the Intelligent Ship Navigation. Wireless Communications and Mobile Computing. 1–9. https://doi.org/10.1155/2022/3437255.

9. Wang, Zhiyuan & Wu, Yong & Chu, Xiumin & Liu, Chenguang & Zheng, Mao. (2023). Risk Identification Method for Ship Navigation in the Complex Waterways via Consideration of Ship Domain. Journal of Marine Science and Engineering. 11. 2265. https://doi.org/10.3390/jmse11122265.

10. Qian, Jingyi & Zeng, Huilu & Yao, Guowei & Kong, Fanwei. (2023). Research of the New Generation Marine Navigation Security Communication System. Transactions on Computer Science and Intelligent Systems Research. 2. 130–139. https://doi.org/10.62051/vvvtye15.

11. Sarkodie, Pokuaa & Zhang, Zhenkai & Benuwa, Ben & Ghansah, Benjamin & Ansah, Ernest. (2018). A Survey of Advanced Marine Communication and Navigation Technologies:

Developments and Strategies. International Journal of Engineering Research in Africa. 34. 102–115. https://doi.org/10.4028/www.scientific.net/JERA.34.102.

12. Yang, Defu & Solihin, Mahmud Iwan & Zhao, Yawen & Yao, Benchun & Chen, Chaoran & Cai, Bingyu & Machmudah, Affiani. (2023). A review of intelligent ship marine object detection based on RGB camera. IET Image Processing. 18. n/a-n/a. https://doi.org/10.1049/ipr2.12959.

13. Jian, Jun & Sun, Zheng & Sun, Kai. (2024). An Intelligent Automatic Sea Forecasting System Targeting Specific Areas on Sailing Routes. Sustainability. 16. 1117. https://doi.org/10.3390/su16031117.

14. Wang, Yong & Gao, Zengyun & Li, Chunxu & Ge, Fan & Wei, Changgeng & Xu, Jiaqing. (2022). Research on Maritime Navigation Perception Requirements of Intelligent Ships. Journal of Physics: Conference Series. 2356. 012033. https://doi.org/10.1088/1742-6596/2356/1/012033.

15. Zhang, Daiyong & Chu, Xiumin & Liu, Chenguang & He, Zhibo & Zhang, Pulin & Wu, Wenxiang. (2024). A Review on Motion Prediction for Intelligent Ship Navigation. Journal of Marine Science and Engineering. 12. 107. https://doi.org/10.3390/jmse12010107.

16. Cui, Zhewen & Guan, Wei & Zhang, Xianku & Zhang, Cheng. (2023). Autonomous Navigation Decision-Making Method for a Smart Marine Surface Vessel Based on an Improved Soft Actor–Critic Algorithm. Journal of Marine Science and Engineering. 11. 1554. https://doi.org/10.3390/jmse11081554.

17. Liu, Qixin & Bai, Xu & Luo, Xiaofang & Yang, Li & Li, Yongzheng & Wang, Ke. (2023). Dynamic Risk Analysis of Intelligent Navigation Process Based on Dynamic Bayesian Network. Journal of Physics: Conference Series. 2491. 012011. https://doi.org/10.1088/1742-6596/2491/1/012011.

18. Du, Yanke & Sun, Shuo & Qiu, Shi & Li, Shaoxi & Pan, Mingyang & Chen, Chi-Hua. (2021). Intelligent Recognition System Based on Contour Accentuation for Navigation Marks. Wireless Communications and Mobile Computing. https://doi.org/10.1155/2021/6631074.

19. Serhii, Firsov & Pishchukhina, Olga. (2018). Intelligent support of multilevel functional stability of control and navigation systems. Radio Electronics, Computer Science, Control. https://doi.org/10.15588/1607-3274-2018-2-20.

20. Zhen, Rong & Ye, Yingdong & Chen, Xinqiang & Xu, Liangkun. (2023). A Novel Intelligent Detection Algorithm of Aids to Navigation Based on Improved YOLOv4. Journal of Marine Science and Engineering. 11. 452. https://doi.org/10.3390/jmse11020452.

21. Luo, Jianping. (2024). Intelligent Stowage Expert Decision-Making System for Ro-Ro Passenger Ships. Electronics, Communications and Networks. https://doi.org/10.3233/FAIA231186.

22. Xue, Xingqun & Ma, Xiaochen & Jiang, Mingnan & Gao, Yang & Park, Sae. (2020). The Construction of an Intelligent Risk-Prevention System for Marine Silk Road. Applied Sciences. 10. 5044. https://doi.org/10.3390/app10155044.

23. Bingchan, Li & Mao, Bo & Cao, Jie. (2018). Maintenance and Management of Marine Communication and Navigation Equipment Based on Virtual Reality. Procedia Computer Science. 139. 221–226. https://doi.org/10.1016/j.procs.2018.10.254.

Пономарьова Вікторія МЕТОД ПІДТРИМКИ ПРИЙНЯТТЯ РІШЕНЬ СУДНОВОДІЯ ДЛЯ АВТОМАТИЗОВАНОГО КЕРУВАННЯ БЕЗПЕКОЮ РУХУ СУДНА ПО ДАНИМ ECDIS

Мета дослідження – підвищення безпеки мореплавства шляхом розробки та застосування методу інтеграції автоматизованих засобів підтримки прийняття рішень (СППР) судноводія в умовах невизначеності. Основною проблемою є високе навантаження на судноводіїв через збільшення обсягу інформації від навігаційних систем, що може призводити до помилок та аварійних ситуацій.

Основна проблема дослідження полягає у необхідності створення автоматизованої системи, яка здатна ефективно фільтрувати великий обсяг інформації та надавати судноводію лише найважливіші дані для прийняття рішень, мінімізуючи ризик помилок у складних навігаційних умовах.

Методика дослідження передбачає створення СППР судноводія на основі великих даних ECDIS та аналітики для ідентифікації суден і аналізу ризиків зіткнень, використовуючи методи розпізнавання

навігаціної інформації для оптимізації маршрутів. Дослідження проводилось на маршруті заходу у порт Лагос, Тимкан, де разом з експертом, було складено словник дій на кожному етапі маршруту, відповідно до даних ECDIS.

Результати дослідження демонструють, що застосування розробленої СППР дозволяє зменшити навантаження на судноводіїв, покращити ситуаційну обізнаність та мінімізувати ризики зіткнень. Зокрема, під час тренажерної підготовки за курсом «Навігація і лоція» з використанням навігаційних тренажерів TRANSAS Wärtsilä Navi-Sailor ECDIS, час руху судна було скорочено від 7% до 18%, що також сприяло економії паливно-мастильних матеріалів та електроенергії на судні.

Практична значущість дослідження полягає в розробці СППР, яка забезпечує фільтрацію та пріоритезацію інформації, покращуючи обізнаність про навігаційну ситуацію. Система використовує методи ідентифікації рівня небезпеки та надає згенеровані автоматичні поради.

Перспективи подальших досліджень включають вдосконалення методів інтеграції різних сенсорів, підвищення точності та надійності навігаційних систем, а також забезпечення кібербезпеки. Розширення дослідження сприятимуть більш глибокій перевірці ефективності запропонованого методу у реальних умовах морської навігації та вдосконаленням алгоритмів аналізу великих обсягів даних.

Бібл. 23, рис. 11.

Ключові слова: *ECDIS; автоматизація; ризик; невизначеність; автоматизовані системи управління; інтелектуальні системи; людський фактор; безпека мореплавства; кваліфікаційні параметри; ідентифікація.*

Статтю прийнято до редакції 26.05.2024