Transportation Research Part E 206 (2026) 104555

Contents lists available at ScienceDirect

Transportation Research Part E

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

Route and speed optimisation of a general cargo ship using extreme gradient boosting and enhanced Deep Q-Network approaches

Yi Zhou [a,*], Serkan Turkmen [b], Kayvan Pazouki [a], Rose Norman [a]

[a] School of Engineering, Newcastle University, Newcastle upon Tyne, UK
[b] Green Maritime Technology Research Group, Tallinn University of Technology, Tallinn, Estonia

ARTICLE INFO

Keywords:
Ship fuel efficiency
Route and speed optimisation
Deep learning
Reinforcement learning
Emission reduction

ABSTRACT

To align with the IMO GHG strategy published in 2023 for reducing CO_2 emissions per transport work, optimising shipping operations is crucial. While applying alternative fuels is the key strategy, their high cost highlights the need to improve operational efficiency in shipping. Route optimisation is one of the key operational measures to reduce fuel oil consumption (FOC) and hence associated emissions. This study presents a novel reinforcement learning-based methodology for route optimization of a cargo ship retrofitted with a Gate Rudder (GR) system, simultaneously targeting FOC reduction, time cost and navigational safety. A foundational aspect of the research is the development of a FOC prediction model using Extreme Gradient Boosting to accurately forecast fuel consumption. The predicted FOC values are then adopted into the environment of an enhanced Deep Q-Network to simultaneously optimise ship speed and route. The results demonstrate that, with the proposed approach, fuel consumption can be reduced by up to 27.81 % compared to the original route operated at service speed prior to the installation of the GR system. Of this reduction, 7.79 % is attributable to the GR system itself, while the remaining 20.02 % results from the proposed route optimization method, considering fuel consumption, voyage time, and safety. This reduction results in a decrease of up to 10,379 kg in CO_2 emissions, which further highlights the environmental benefits of the proposed optimisation approach, as well as the GR system.

1. Introduction

Reducing emissions from the marine industry is crucial, as shipping activities significantly contribute to global greenhouse gases and harmful emissions, affecting both local and global environments (Gibson et al., 2019). To achieve the targets established by the International Maritime Organization (IMO), the international shipping sector must aim for reducing CO_2 emissions per transport work, as an average across international shipping, by at least 40 % by 2030, compared to 2008 (MEPC, 2023) Numerous studies have explored strategies to minimize fuel consumption and, consequently, emissions (Endresen et al., 2003; Eyring et al., 2010; Walsh and Bows, 2012; Opdam & Bijlard, 2024). Fuel consumption and hence GHG emissions can be reduced by, for example fuel usage data analysis (Trodden et al., 2015), efficient propulsion systems (Turkmen et al. 2023), e-fuels (Lindstad et al., 2021), clean fuels (Hoang

* Corresponding author at: School of Engineering, Newcastle University, Newcastle upon Tyne, UK.
E-mail address: nyz51@newcastle.ac.uk (Y. Zhou).

https://doi.org/10.1016/j.tre.2025.104555
Received 23 February 2025; Received in revised form 8 October 2025; Accepted 12 November 2025
Available online 20 November 2025

et al. 2022; Hoang et al. 2023), or operational and route optimisation (Moradi et al., 2022). One of the emission reduction technologies contributing to efficient propulsion systems is the Gate Rudder system, recently retrofitted on a commercial vessel to improve power-speed performance (see Fig. 1). The Gate Rudder (GR) system is an Energy Saving Device (ESD) installed as part of a ship propulsion system, which is made up of a twin rudder setup with two asymmetric section blades positioned on either side of the propeller. The GR enhances the flow around the propeller by inducing axial velocity in the propeller plane, which generates additional thrust and helps recover viscous resistance losses by equalizing the ship's wake, thus enhancing the propulsive efficiency (Sasaki & Atlar, 2018; Turkmen et al. 2023). In addition to its energy-saving capabilities, the GR system provides superior steering performance, especially at low speeds, by enabling unique manoeuvring modes such as "crabbing," which facilitates lateral movement during berthing. Moreover, the system also reduces flow-induced vibration and noise and allows for more flexible hull and engine room designs due to its forward positioning relative to conventional rudders (Carchen et al., 2021).

To meet the IMO target for GHG reduction by 2030, efforts must be made across all facets of the shipping industry. Although the use of ESDs offers a practical approach to reducing ship emissions, there remains significant potential for further emission reductions through route and speed optimisation (Moradi et al., 2022).

For the effective implementation of this system, a precise FOC prediction model is essential for route planning, optimisation, and assessment (Papandreou and Ziakopoulos, 2022). Three primary approaches, deterministic methods (Tillig F and Ringsberg JW, 2019; Simonsen M et al. 2018), data-driven techniques (Gkerekos et al., 2019; Uyanık et al., 2020), and hybrid approaches that combine elements of both (Coraddu et al., 2017; Zhou et al. 2022), have been adopted in the recent literature. Fan et al. (2022) highlighted in their review that for applications demanding high accuracy, such as the FOC prediction model in this study, data-driven models are the more appropriate choice. These models are typically developed in a single-stage format, functioning as "black boxes" trained using vessel operational and Metocean data from onboard sensors, which has become increasingly feasible with the advancement of Internet of Things (IoT) technologies and IoT-enabled sensors (Chung et al., 2025). Over the past decade, various data-driven methodologies, including Artificial Neural Networks (ANNs) (Beşikçi et al., 2016; Karagiannidis et al., 2019), Support Vector Machines (SVM) (Gkerekos et al., 2019), Random Forest (RF) (Uyanık et al., 2020; Chen et al., 2023; Zhou et al., 2023), Multivariate Polynomial Regression, and Extreme Gradient Boosting (XGBoost) (Papandreou and Ziakopoulos, 2022), have been utilized for FOC estimation. Among these machine learning methods, XGBoost tends to outperform other models like RF and ANNs in terms of training speed and efficiency while providing comparable accuracy. More recent work focused on deep learning methods using a Long Short-Term Memory (LSTM) based model to estimate FOC (Zhang et al., 2024), which necessitated additional steps to pre-process the input data into a 3D format, arranged in a sequence that aligned with the requirements of the model. The above supervised learning methods could serve as complementary tools to provide necessary information related to FOC usage for route optimisation purposes.

In the realm of marine route optimisation, a variety of methods have been developed and refined over the years to enhance operational efficiency and reduce fuel consumption. Table 1 provides an overview of the methods that can be found in literature. Traditional optimisation techniques have played a significant role in this field, with several well-established methods being frequently employed. However, traditional methods, such as Dijkstra's algorithm (Mannarini et al., 2024) and A* search (Kaklis et al., 2024), always struggle with real-time adjustment. Isochrone methods, which divide the ocean into a grid and use weather forecasts to determine the optimal path for minimizing travel time or maximizing safety, lack the ability to account for vessel speed variations. This limitation reduces their effectiveness when it comes to optimizing fuel consumption. Moreover, these methods are generally combined with straightforward weather routing techniques that do not dynamically adjust the route based on real-time conditions (Hagiwara, 1989; Wang et al., 2019).

To overcome these drawbacks, more advanced Intelligent Search Algorithms and Mathematical Programming methods have emerged, for instance, Dynamic Programming, Genetic Algorithm and Particle Swarm optimisation (PSO). Dynamic Programming breaks down the optimisation problem into smaller, more manageable subproblems, which are then solved recursively. However,

Fig. 1. A retrofitted GR System Configuration.

Table 1
Methods Applied in Ship Route Optimisation.

Methods	Specific Techniques	References
Traditional Algorithms		
	Dijkstra	Mannarini et al. (2024)
	A* Search	Kaklis et al. (2024)
	Isochrone Methods	HAGIWARA (1989), Lin et al. (2013)
Intelligent Search Algorithms and Mathematical Programming	Dynamic Programming	Wang et al. (2021), Calvert (1990),Zaccone et al. (2018) and Wang et al. (2019)
	Genetic Algorithm	Vettor and Guedes Soares (2016), Lee et al. (2018) and Li et al. (2024)
	PSO	Du et al. (2022) Du et al. (2023)
	Kwon Method	Shao & Zhou (2011), Lu et al. (2015)
Reinforcement Learning	ANN + DDPG	Moradi et al. (2022)
	PPO	Zhu et al. (2024)

dynamic programming requires a comprehensive understanding of the environment, which might result in high computational costs and reduced flexibility, particularly when adapting to the dynamic nature of marine conditions (Calvert, 1990; Wang et al., 2021). Techniques such as Genetic Algorithms are designed to mimic the processes of natural evolution to find optimal solutions. These algorithms are particularly well-suited for solving complex, nonlinear, and multi-modal optimisation problems. They allow for the exploration of a broad range of potential routes and the selection of the most efficient one based on multiple criteria, including fuel consumption, safety, and travel time. Other techniques like PSO, harness swarm intelligence to solve the multi-objective optimisation problem, and have attracted significant research interest due to their effectiveness in global search and optimization. However, traditional PSO often struggles with local optima and slow convergence when tackling high-dimensional real-world problems (Yao et al., 2024). A common limitation of these searching and programming algorithms is their slow convergence rate, which often demands substantial computational resources. Additionally, these algorithms require careful parameter tuning, which can be both challenging and time-intensive (Marie and Courteille, 2009; Lee et al., 2018; Vettor and Guedes Soares, 2016; Du et al., 2023).

Reinforcement Learning (RL) has emerged as a promising and innovative approach in the optimisation of marine routes. Unlike traditional methods, RL is capable of learning optimal strategies through continuous interaction with the environment. Techniques such as Deep Q-Network (DQN) and Deep Deterministic Policy Gradient (DDPG) have demonstrated significant potential. In terms of its application in marine route optimisation, although the DDPG model outperformed the DQN model in the research of Moradi et al. (2022), for ship route and speed optimisation during transit periods, it is important to note that frequent adjustments to the ship's course and speed based on DDPG optimisation are often inappropriate in many practical scenarios during ship transit mode. Additionally, achieving the optimisation strategies derived from continuous action models like DDPG requires more precise control techniques for the propulsion system, which could impose additional requirements and incur extra costs to update the on-board control system. From an RL modelling perspective, DDPG tends to require a greater number of timesteps to achieve convergence compared to other models (Henderson et al., 2018). The algorithm's reliance on exploration noise can lead to sudden failures, particularly in unstable environments, making accurate Q-value estimation of expected returns challenging. This difficulty is compounded by the fact that many exploratory trajectories may result in failure. Furthermore, DDPG typically demands a substantial number of training episodes to discover effective solutions (Lillicrap et al., 2015), which can limit its applicability due to the significant computational resources required. Conversely, DQN is particularly suited to environments with discrete action spaces because it directly approximates Q-values for each action, allowing for efficient learning and generalization across different states (Mnih et al., 2015). Moreover, DQN's performance can be further enhanced with techniques like Prioritized Experience Replay (Hessel et al., 2018), Double Q-Learning (van Hasselt et al., 2016), Dueling Networks (Wang et al., 2016), and Multi-Step Learning (Hessel et al., 2018). These methods provide benefits such as improved data efficiency, reduced overestimation bias, better generalization, and quicker propagation of newly observed rewards, respectively, and ultimately leading to better overall learning performance over traditional methods, intelligent search algorithms and mathematical programming approaches. With a well-defined grid for ship routes, these enhanced DQN methods can be effectively applied to optimise route planning.

Although numerous studies have explored the application of RL methods in the marine field, certain research gaps remain. For example, Chen et al. (2019) considers collision with obstacles but the costs during the voyage, such as FOC, are not considered. Zhu et al. (2024) applied a RL method to optimise the efficiency of a fishing vessel by setting objectives to maximize catch, while minimizing distance and risk. Nevertheless, a generic FOC model was absent in their approach to estimate ship FOC in various operation, loading and weather conditions during the trips. As a result, FOC was not included in the cost function for maximising the catch. Moradi et al. (2022) investigated the use of RL for optimising ship routes with objectives focused on reducing fuel consumption and wave height along the voyage. They also imposed time constraints for travel in some scenarios, which could not be exceeded. However,

the obstacles the ship might encounter during the voyage, such as islands, were omitted in their research. This omission limits the applicability of their approach to regions with numerous geographical obstacles, such as the Tyrrhenian Sea and the Mediterranean Sea. A more comprehensive investigation is needed to adapt RL methods to route optimisation, which takes into account fuel usage, safety, time limitations, and geographical constraints.

To address these research gaps, this study proposes a novel methodology for route optimisation using RL to optimise the operation of a general cargo ship in the Tyrrhenian Sea. The core of the research involves developing a generic ship model using XGBoost to accurately forecast FOC. The predicted FOC values are then integrated into the environment of an enhanced version of the DQN, which incorporates advanced techniques for more efficient learning than standard DQN. This approach optimises ship operations by considering FOC, safety, voyage time, and geographical limitations. To assess the performance insights of the proposed approach, its performance is benchmarked against A* Search (as a traditional algorithm) and PSO (representing intelligent search methods) in terms of optimisation performance and computational cost within the same navigation environment. It is also important to note that the ship has been retrofitted with a GR system. In addition to exploring fuel savings from RL-based route and speed optimisation, this study also highlights the potential fuel savings resulting from the use of the GR system for the target voyage.

The remainder of this paper is structured as follows: **Section 2** introduces the methodology, which provides a detailed overview of the mathematical foundations of the implemented methods, as well as the strategies for data collection, processing and the generic FOC model development. Additionally, it elaborates on the theoretical framework for the development of the RL-based route and speed optimisation model, as well as the A* search and PSO models, which are applied to provide the baseline comparisons. **Section 3** presents the generic FOC model's performance during both validation and testing phases. In addition, scenarios are designed for the RL based route and speed optimisation models. The results and comparisons are provided to demonstrate how the proposed approach can effectively optimize ship operations when specific objectives are applied. Finally, **Section 4** offers concluding remarks.

2. Methodology

Fig. 2 presents the overall workflow of the methodology developed in this paper. The process begins with the FOC Model Development, where data collection and preprocessing are conducted to gather relevant information needed for accurate FOC prediction. This stage involves training the model using the prepared data and subsequently validating its performance on unseen test datasets to ensure accuracy and generalizability. Following this, the RL Environment Setup is established, where key parameters such as weather and ship loading conditions, inaccessible route obstacles (islands and geographical boundaries), FOC rates based on the developed FOC model, and time constraints are defined. This step also involves specifying the observation space, action space, and reward function, which are essential components for the RL agent's learning and decision-making processes. Finally, the workflow proceeds to RL Agent Training for Route Optimisation, where the RL agent is trained with specific parameters, including episode number, batch size, and learning rate. The agent's training focuses on optimising ship routes by evaluating and comparing different strategies to achieve the optimisation objectives.

2.1. FOC model development

The methodology outlined in this work is illustrated in Fig. 3 and comprises several key stages: a) Data collection for the target ship,

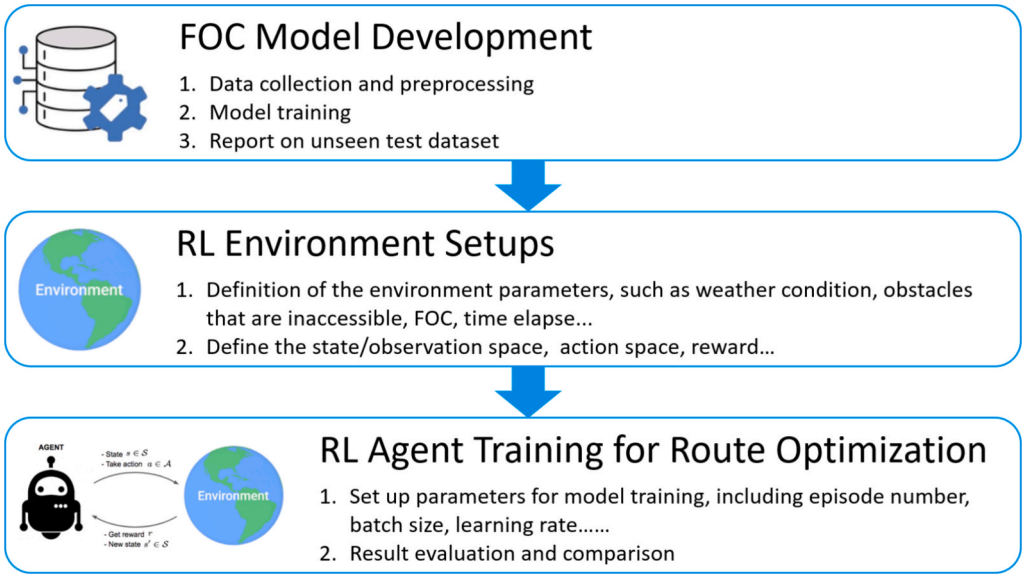


Fig. 2. Overall Workflow of the Proposed Methodology.

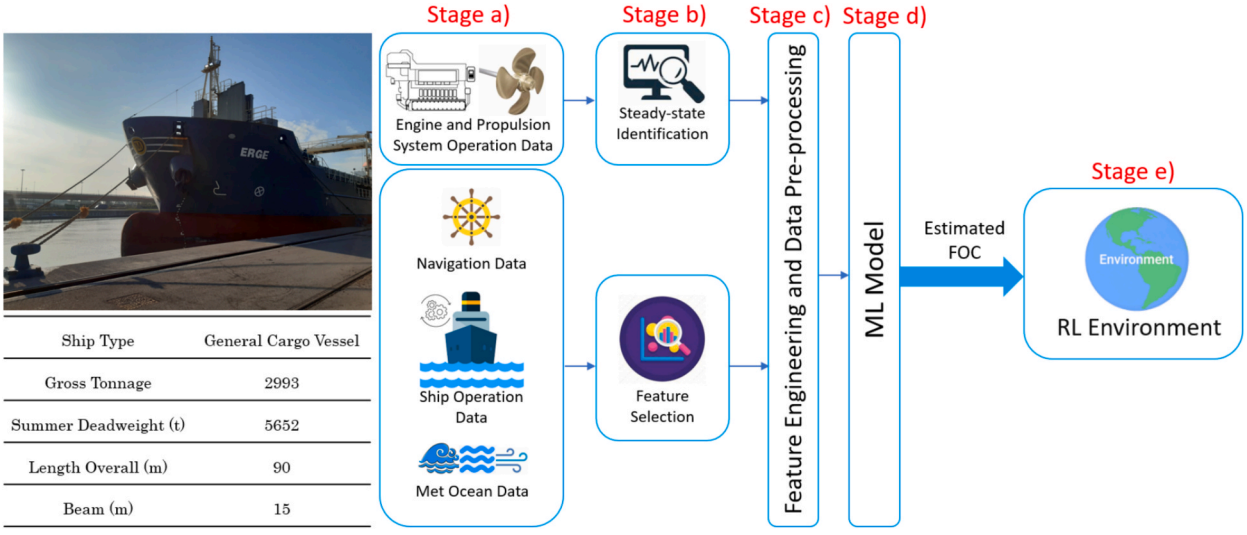
which gathers data pertaining to engine and propulsion system operation, navigation and ship operation from sensors installed on a general cargo vessel, through a performance monitoring system. This data set also includes weather data obtained from open sources such as the Copernicus Marine Environment Monitoring Service (CMEMS), given the absence of weather-related sensors on the vessel. b) The process of feature selection and engine steady-state data identification is conducted to identify relevant features and ensure the dataset reflects steady-state operating conditions, which are typically associated with ship transit periods, where the vessel maintains a relatively constant course and speed. Given that the objective of this study is to optimise ship routing and speed, only data corresponding to these transit periods are considered. c) During the feature engineering and data preprocessing stage, data is prepared for model development. This involves comprehensive data cleaning to identify and exclude engine transients and recorded anomalies, ensuring the data reflects steady-state operation. The units of variables are standardized to maintain consistency, and feature standardization is performed to promote faster convergence during model training, resulting in a dataset optimally formatted for modelling. d) A Machine Learning (ML) model is then developed to estimate the FOC of the target ship. e) Finally, the developed ML model provides FOC information to the RL environment, serving as a critical input for the subsequent RL modelling phase.

2.1.1. Data Acquisition

The data utilized in this study was collected on board using the CETENA Performance Monitoring system (Shaw & Lin, 2021), which provides data samples at one-minute intervals. This system includes a PC with specialized software designed to record all available onboard data, as well as hardware dedicated to acquiring signals related to propulsion efficiency, such as torque, shaft power, and FOC. Additionally, the monitoring system is connected to the integrated navigation system to gather the following data: date, time, latitude, longitude, speed over ground, and course from the GPS. Ship deck officers are responsible for manually recording additional voyage-specific information—such as displacement, draft forward, and draft aft—into a spreadsheet and the "Repeater" software provided by CETENA. In this research, the Metocean data were obtained from the CMEMS. Specifically, these variables were from the following short-term forecast products: a) CMEMS Global Ocean 1/12° Physics Analysis and Forecast updated Daily (PHY) (Global Monitoring and Forecasting Center, 2019a) for ocean current variables; b) CMEMS Global Ocean Waves Analysis and Forecast (WAV) (Global Monitoring and Forecasting Center, 2019b) for wave variables; and c) CMEMS Global Ocean Hourly Sea Surface Wind and Stress from Scatter meter and Model (WIND) (Global Monitoring and Forecasting Center, 2022).

2.1.2. Feature selection

Drawing on domain expertise in FOC modelling, it is evident that Metocean parameters such as wave height, wind speed, current speed, and their absolute directional values are crucial for accurate FOC modelling. Additionally, insights from prior studies (Gkerekos et al., 2019; Gkerekos & Lazakis, 2020; Xie et al., 2023; Zhang et al., 2024) further reinforce the importance of these parameters in enhancing the precision of FOC models. Consequently, Metocean-related variables, including relative sea current velocity, relative wind speed and direction, and significant wave metrics (Height and direction), are considered as the inputs for the FOC model development. This model generates essential FOC data to support the subsequent route and speed optimization phase. Additionally, during maritime operations, critical navigational data such as ship speed over ground and course, in conjunction with these environmental conditions, play a vital role in the efficient management of the vessel's energy and navigational systems. These navigational parameters are defined as the action space for the RL modelling discussed in **Section 2.2**. Moreover, ship displacement and draft (both forward and after) are incorporated as input features, as they provide important information about the ship's loading condition in terms of cargo. These factors have been consistently employed in previous FOC modelling studies (Gkerekos et al., 2019; Gkerekos & Lazakis, 2020). Therefore, these variables have been selected as primary inputs for the training phase of the FOC model.



Ship Type	General Cargo Vessel
Gross Tonnage	2993
Summer Deadweight (t)	5652
Length Overall (m)	90
Beam (m)	15

Fig. 3. Ship generic model development for FOC Estimation.

2.1.3. Engine Steady-State mode identification

Main engine idle periods are defined by conditions where the shaft RPM is below 70 and the SOG is less than 4 knots. Previous research by Castresana et al. (2023) identified engine speed and the Fuel Oil Injection Pump Rack position (FORACK) as crucial parameters for classification. However, in this study, direct readings of engine RPM and FORACK are not available; therefore, shaft RPM is used to identify steady-state engine operation. Although there may be slight delays due to inertia between engine RPM and shaft RPM, these are typically minimized in well-designed marine propulsion systems. The study employs the Relative Standard Deviation (RSD) over a 10-minute window preceding each data point, as recommended by MEPC (2008), applying it to shaft RPM values. The calculation of RSD, as shown in **Eq. (1)**, involves determining the standard deviations and moving averages for the 10 min preceding each sample.

$$RSD_{10min} = \frac{Sd_{10min}}{\overline{x_{10min}}} \times 100\% = \frac{\sqrt{\frac{\sum_{i=1}^{n}(x_i - \overline{x_{10min}})^2}{n-1}}}{\overline{x_{10min}}} \times 100\% \tag{1}$$

In this context, Sd_{10min} represents the standard deviation over the previous 10 min for each sample, while $\overline{x_{10min}}$ denotes the moving average calculated over the same period. The variable x_i refers to the i^{th} sample observation, and n is the number of samples of the 10-minute window preceding each sample. To ensure that each sample reflects a relatively steady state, a threshold of $RSD_{10min,RPM} \leq 5\%$ was set. Samples exceeding this threshold are classified as non-steady-state engine activity and are consequently excluded from further analysis.

2.1.4. Data preprocessing and feature engineering

In **Stage c)** of Fig. 3, the initial step involves removing undesirable data samples resulting from sensor failures or monitoring system issues, such as NaN values and unrealistic readings. Additional filtering is applied to exclude data from engine idle or transient modes according to the steady-state mode identification method. These methods have been applied to ensure that the data are healthy and address the uncertainty of measurement. Furthermore, the retrieved weather data from CMEMS undergo further processing to prepare them for the subsequent modelling stage. This section will provide a detailed overview of these preprocessing steps.

- **Copernicus Data Processing**

CMEMS offers user interfaces for extracting data based on the vessels' position and datetime indexes. To align data extraction with these indexes, the ship's position and time data, provided for each minute, must be matched with the three indexes from the environmental data sources. The Nearest Neighbours Imputation method, as proposed by Faisal and Tutz (2021), is employed to identify the first nearest neighbouring indexes for retrieving environmental data. For data retrieval, such as matching environmental data to specific ship positions and times, the method involves identifying the nearest data points in the source dataset to the target points in the query dataset. The retrieved metocean data is then transformed to a vessel-based fixed frame using the method proposed by Gkerekos & Lazakis (2020).

- **Feature Engineering**

Additionally, z-score normalization is applied before model training to improve convergence speed, as suggested by Ioffe and Szegedy (2015). The process involves calculating the mean, μ_i, and standard deviation, σ_i, derived from variable x_i. Then the standardized variable, $x_{standardized}$, can be obtained from **Eq. (2)**:

$$x_{standardized} = \frac{x_i - \mu_i}{\sigma_i} \tag{2}$$

Eq. (2) is applied to the training and validation datasets to determine the scaler, which is subsequently used to standardize the testing datasets.

2.1.5. FOC modelling algorithms

While time series modelling approaches like LSTM-based and GRU-based methods can provide high accuracy by considering hundreds of previous states, this approach can dramatically increase computational demands when integrated into real-time route optimisation environments. Moreover, complex data preprocessing is required for the input data for these models, typically, in a 3D format: [samples, time steps, features]. The time step dimension represents the length of the sliding window, a technique that segments historical data into overlapping sequences for model training. In order to exclude the undesirable data caused by sensor failures, and that collected during ship manoeuvring, preliminary data cleaning has been applied in this work. However, this introduces gaps in timestamps, resulting in non-continuous periods that are not favourable for temporal sequence learning. Although there are some methods to address these data gaps (Zhou et al., 2025), this results in the loss of data samples used for model development, which means they are not preferable for the relatively small dataset applied in this work. XGBoost is applied in this work to maintain consistency with methodologies from similar work (Xie et al., 2023), which also utilized XGBoost-based FOC modelling for ship optimization purposes. The XGBoost algorithm offers greater computational efficiency due to its faster development time and ease of implementation. A more detailed overview of the algorithm is presented in Chen and Guestrin (2016). It integrates multiple learners

internally and aims to minimize the objective function shown in **Eq. (3)**:

$$L(\varphi) = \sum_{i=1}^{I} l(\hat{y}_i, y_i) + \sum_{m=1}^{M} \Omega(f_m)$$
$$\text{where} \quad \Omega(f) = \tau L_T + \frac{\lambda}{2}\|c\|^2 \tag{3}$$

Here φ is the prediction function of the tree ensemble model. l is the loss function applied to measure the difference between the predicted target value $\hat{y}_i$ and the true target value y_i. f_m is the m-th regression tree in the ensemble of M trees, with tree structure q and leaf weights c. Furthermore, a penalizing term, Ω, is incorporated to control model complexity, where τ, λ are penalizing coefficients. L_T denotes the total number of leaves in the regression tree. The tree ensemble model in **Eq. (3)** involves functions as parameters and cannot be optimized by applying conventional optimisation methods in Euclidean space. Specifically, at the z-th iteration, the prediction for the i-th instance denoted as $\hat{y}_i^{(z)}$, an additional function f_z is added to minimize the following objective to drive the gradient boosting process.

$$L^{(z)} = \sum_{i=1}^{I} l\left(\hat{y}_i^{(z-1)} + f_z(x_i), y_i\right) + \Omega(f_z) \tag{4}$$

Here x_i represents the feature vector for the i-th instance in the dataset. Second-order approximation can be applied to optimize the objective:

$$L^{(z)} \simeq \sum_{i=1}^{I} \left[l\left(\hat{y}_i^{(z-1)}, y_i\right) + g_i f_z(x_i) + \frac{1}{2} h_i f_z^2(x_i) \right] + \Omega(f_z) \tag{5}$$

where g_i and h_i represent the first and second derivative of the loss function, respectively. When XGBoost iteration generates a new regression tree, the constant term in **Eq. (5)** can be omitted:

$$\tilde{L}^{(z)} = \sum_{i=1}^{I} \left[g_i f_z(x_i) + \frac{1}{2} h_i f_z^2(x_i) \right] + \Omega(f_z) \tag{6}$$

Given $I_j = \{i | q(x_i) = j\}$, where I_j represents the set of instances assigned to the j-th leaf of the tree based on the tree structure $q(x)$. Using this definition, **Eq. (6)** can be rewritten by expanding the penalizing term Ω as follows:

$$\tilde{L}^{(z)} = \sum_{j=1}^{L_T} \left[\left(\sum_{i \in I_j} g_i \right) c_j + \frac{1}{2} \left(\sum_{i \in I_j} h_i + \lambda \right) c_j^2 \right] + \tau L_T \tag{7}$$

The optimal weight c_j^* of the leaf j can be calculated as $c_j^* = -\frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \lambda}$ for a fixed tree structure. XGBoost includes a phase dedicated to hyperparameter tuning. This process aims to optimize the model's performance by adjusting various parameters. In this work, some of the key hyperparameters are tuned for this purpose, including the number of estimators, the maximum depth of each decision tree and the learning rate.

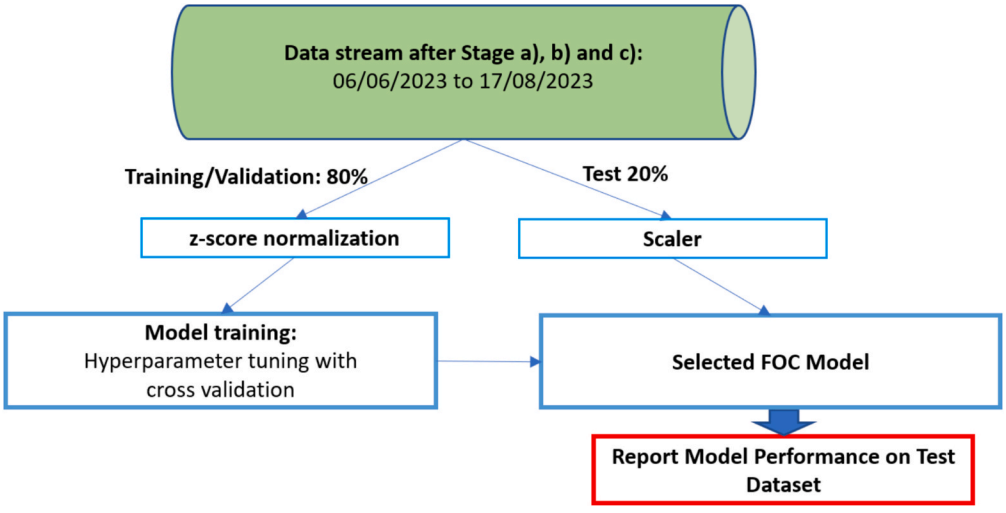


Fig. 4. FOC Model Development.

2.1.6. FOC model development

This section will introduce the model development as shown in Fig. 3 (**Stage d**). As, illustrated in Fig. 4, the data collected over the period from 06/06/2023 to 17/08/2023 from the target ship are applied to develop the FOC model. After data pre-processing, the data stream will be partitioned by randomly splitting the datasets into 80 % for training/validation and 20 % for testing, respectively. The z-score normalization method is applied to the training and validation datasets to obtain the scaler, which is then later used to scale the testing datasets. During the model training phase, hyperparameter optimisation was performed. This process identified the optimal hyperparameter sets based on cross-validation error. The performance of the model was then evaluated on the test dataset.

In this study, the K-fold cross-validation is applied for model development and hyperparameter tuning to ensure that selected hyperparameter sets approach optimal and are not just overfitting the model (Wieczorek et al., 2022; Xie et al. 2023). Fig. 5 introduces the K-fold cross-validation strategy, where the dataset is divided into K splits, each containing K folds. During the cross-validation process, the training dataset is partitioned into K equally sized folds, and K iterations of training and validation are conducted. In each iteration, one segment k is set as the validation set, while the remaining $K-1$ segments are used for model training. This strategy can be further implemented for hyperparameter tuning. Specifically, a total of K runs are performed for each of the S hyperparameter sets during the cross-validation stage, facilitating a thorough assessment of model performance across different configurations. The optimal hyperparameter set $n_{s,opt}$ is determined based on the errors $[e_{n_1}, \cdots, e_{n_S}]$. Finally, the model is retrained on the whole training dataset given the optimal hyperparameters.

2.1.7. Model evaluation metrics

The key indicator of model evaluation in this work in cross-validation is Mean Absolute Error (MAE), presented as (Papandreou and Ziakopoulos, 2022):

$$MAE(y_i, \hat{y}_i) = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{8}$$

where n is the number of samples in y, $\hat{y}_i$ is the predicted value, and y_i is the true value. $|y_i - \hat{y}_i|$ calculates the absolute error (AE) over n samples. This is applied as the performance metric in the validation stage.

In machine learning, R^2 is often used to evaluate the performance of a regression algorithm. It can be expressed as:

$$R^2 = 1 - \frac{\sum_i(\hat{y}_i - y_i)}{\sum_i(\overline{y_i} - y_i)} \tag{9}$$

An R^2 value closer to 1 indicates that the regression algorithm predicts the target variable with higher accuracy. The R^2 metric is instrumental in evaluating the goodness of fit for a regression model, providing insight into the model's performance. It is particularly useful for comparing the efficacy of multiple models, identifying the best-performing model, and determining which factors most significantly impact model performance during the optimisation process, thereby guiding targeted improvements. According to Chicco et al. (2021), R^2 is both informative and truthful, without the interpretability limitations associated with other metrics.

A variant of MAE is Mean Absolute Percentage Error (MAPE), expressed as:

$$MAPE(y_i, \hat{y}_i) = \frac{1}{n}\sum_{i=1}^{n}\frac{|y_i - \hat{y}_i|}{y_i} \times 100\% \tag{10}$$

In practice, a major drawback of MAPE is that it becomes numerically unstable when there exists an i such that $y_i = 0$ (Gkerekos et al., 2019). However, there are few samples with target values close to 0 in this work as engine idle periods are filtered out at the data pre-

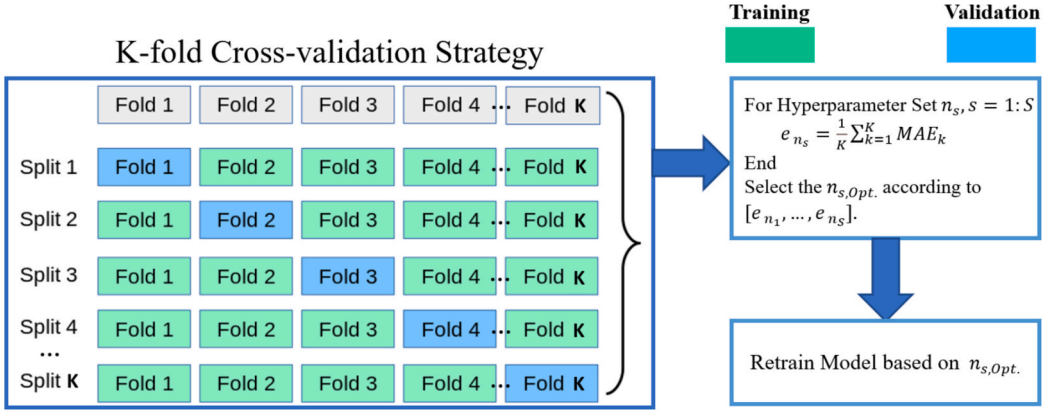


Fig. 5. K-fold Cross-validation Strategy with Hyperparameter Tuning.

processing stage. Thus, this metric is adoptable in this case. R^2 and MAPE are applied to evaluate the model performance in terms of precision on the test dataset.

2.2. Rl-based Optimisation: DQN model

Fig. 6 illustrates the working principle of the DQN model in this work. For discrete time step $t = 0, 1, 2\cdots$, the environment provides an observation S_t to the agent, which subsequently selects an action A_t. The environment then delivers the next state, reward, and discount, following the framework of a Markov Decision Process (MDP). An MDP can be represented as a tuple $\langle \boldsymbol{S}, \boldsymbol{A}, T, r, \gamma \rangle$, where $\boldsymbol{S}$ is a set of states, $\boldsymbol{A}$ denotes a set of actions, $T(s, a, s') = P[S_{t+1} = s' | S_t = s, A_t = a]$ denotes the transition function, $r(s, a) = \mathbb{E}[R_{t+1} | S_t = s, A_t = a]$ represents the reward function, and γ is the discount factor, which ranges from 0 to 1. The agent's action selection is governed by a policy π that defines a probability distribution on actions for each state. Given a state S_t at time t, the discount return is defined as $G_t = \sum_{k=0}^{\infty} \gamma_t^{(k)} R_{t+k+1}$, which is retuned as the discounted sum of future rewards collected by the agent, where the discount in the future for a reward k steps is presented by $\gamma_t^{(k)} = \prod_{i=1}^{k} \gamma_{t+i}$. The objective of the agent is to maximize the expected discounted return by discovering an optimal policy. The policy could be directly learned or inferred as a function of other learned quantities. An estimate of the expected discounted return, or value in value-based reinforcement learning could be learned by the agent, when aligning with a policy π commencing from a given state, $v^{\pi}(s) = E_{\pi}[G_t | S_t = s]$, or from state-action pair, $q^{\pi}(s, a) = E_{\pi}[G_t | S_t = s, A_t = a]$. A common way of obtaining a new policy from a state-action value function is to apply the ϵ-greedy approach based on the action values, which involves selecting the action with the highest value (the greedy action) with a probability of $(1 - \epsilon)$, or choosing an action uniformly at random with a probability of ϵ. These policies are applied to introduce a form of exploration, which selects actions in a random form that are sub-optimal based on its current estimation, then the agent can observe and improve its estimates.

2.2.1. Standard DQN

In deep reinforcement learning, there are various components of agents, such as policies $\pi(s, a)$ or value $q(s, a)$, applied in deep neural networks training process to minimize the loss function. In DQN, deep networks and reinforcement learning ideally work with each other by applying a convolutional neural network to estimate the action values for a given state s_t. The agent continuously selects an action using the ϵ-greedy strategy according to the action values and stores a transition $(S_t, A_t, R_{t+1}, \gamma_{t+1}, S_{t+1})$ to a replay memory buffer. By applying stochastic gradient descent to minimize the below loss function, the parameters of the neural network are optimised:

$$L = (R_{t+1} + \gamma_{t+1} max_{a'} q_{\overline{\theta}}(S_{t+1}, a') - q_{\theta}(S_t, A_t))^2 \tag{11}$$

where t is a time step picked from the replay memory randomly. The loss gradient is propagated exclusively through the online network's parameters θ, while the term $\overline{\theta}$ denotes the target network's parameters. Specifically, A periodic replica of the online network isn't directly optimized. Instead, RMSprop, a variant of stochastic gradient descent, is used for optimization. Thus, for the loss defined in **Eq. (11)**, the time index t is chosen randomly from previous transitions. The adoption of experience replay and target networks could increase the stability of learning Q values.

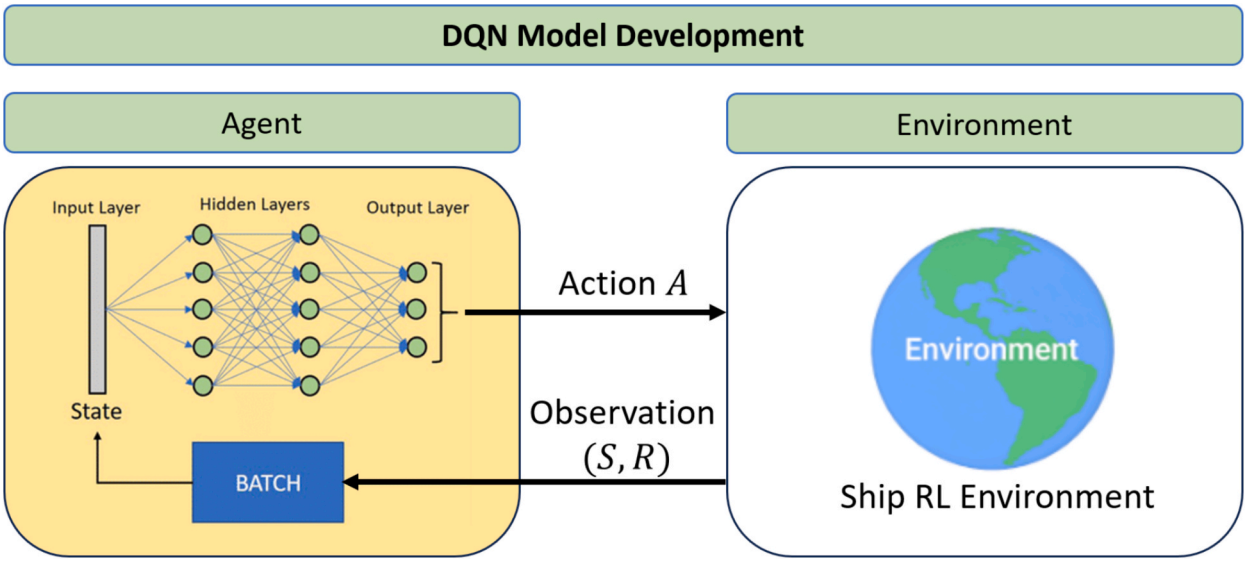


Fig. 6. Framework of DQN Model.

2.2.2. Enhanced DQN implement

To boost the performance of standard DQN, several advanced features have been applied in this work, including Prioritized Experience Replay, Double DQN, Dueling Network Architecture and Multi-Step Learning:

- **Prioritized Experience Replay (PER)**

DQN samples are uniformly collected from the replay buffer. Generally, it is more favourable to sample more frequently those transitions which are worthy to learn. The prioritized experience replays samples transitions with probability of p_t when given the last absolute Temporal Difference error encountered:

$$p_t \propto |R_{t+1} + \gamma_{t+1} max_{a'} q_{\bar{\theta}}(S_{t+1}, a') - q_\theta(S_t, A_t)|^\omega \tag{12}$$

where ω is a hyperparameter that impacts the shape of the distribution. New transitions are added into the replay buffer with highest priority, which exhibits a preference for the latest transitions. It is worth noting that stochastic transitions might also be beneficial, even when there is not much left to learn about them.

- **Double DQN**

Traditional Q-learning is influenced by the bias of overestimation, which is caused by the maximization step in **Eq. (11)**. This can impact the learning process. Double Q-learning mitigates overestimation by separating the process of selecting an action from evaluating it. This decoupling can be effectively integrated into DQN by incorporating it into the loss function:

$$L_{DDQN} = (R_{t+1} + \gamma_{t+1} q_{\bar{\theta}}(S_{t+1}, argmax_{a'} q_\theta(S_{t+1}, a')) - q_\theta(S_t, A_t))^2 \tag{13}$$

This modification was observed to mitigate the harmful overestimations present in DQN, thereby enhancing performance.

- **Dueling Network Architecture**

The dueling network is a neural network structure specifically engineered for value-based reinforcement learning. Generally, it involves two streams of computation, which are named value and advantage streams, and are merged by a unique aggregator. The following factorization of action values is required:

$$q_\theta(s, a) = v_\eta\left(f_\xi(s)\right) + a_\psi\left(f_\xi(s), a\right) - \frac{\sum_{a'} a_\psi\left(f_\xi(s), a'\right)}{N_{actions}} \tag{14}$$

Where ξ, η and ψ are the parameters of the shared encoder f_ξ, the value stream v_η, and the advantage stream a_ψ, respectively. The concatenation of them can be presented as $\theta = \{\xi, \eta, \psi\}$.

- **Multi-Step Learning**

Q-learning sums up a single reward and then the greedy action is applied at the next step for bootstrapping. Forward-view multi-step targets can also be applied as an alternative. The truncated *n*-step return from a given state S_t could be defined as:

$$R_t^{(n)} \equiv \sum_{k=0}^{n-1} \gamma_t^{(k)} R_{t+k+1} \tag{15}$$

By minimizing the alternative loss function, one can define a multi-step variant of DQN to accelerate the learning process.:

$$L_{MDQN} = \left(R_t^{(n)} + \gamma_t^{(n)} max_{a'} q_{\bar{\theta}}(S_{t+n}, a') - q_\theta(S_t, A_t)\right)^2 \tag{16}$$

These independent techniques are incorporated into the standard DQN model in this study to improve its learning efficiency within the context of a marine route optimisation environment. In **Section 3.2**, the performance of the standard DQN will be compared with the enhanced DQN, which includes these techniques, to highlight their advantages and effectiveness in optimising marine routes.

2.2.3. State and observation space

A fundamental requirement for RL environments is their ability to effectively connect the agent's learned knowledge to its current situation. This requirement, known as the Markov property, ensures that the present state encapsulates all the necessary information, making it sufficient for the agent to predict future states without needing additional context. Essentially, this implies that the current state must fully reflect all pertinent details from previous steps. Furthermore, it is crucial not to overwhelm the agent with excessive information, as this can lead to confusion by complicating the association between states and the rewards received. Thus, carefully selecting the observation space is vital for RL modelling. In this study, the agent's observation space is defined by geographical location (latitude and longitude) and time. The inclusion of time as an additional parameter allows the agent to capture dynamic, time-dependent weather conditions. Combined with location, this enables the retrieval of relevant weather data and the prediction of

FOC for a given ship SOG and COG, thereby providing cost-related feedback to guide the learning process of optimisation models. Accordingly, the observation space incorporates 13 variables: ship location (latitude and longitude), time, weather conditions (wind speed and direction, wave height and direction, current speed and direction), ship loading condition (draft forward, draft aft, and displacement), and the FOC predicted by the XGBoost model. This ensures the agent has sufficient information for more effective and robust decision-making.

2.2.4. Action space

To implement the DQN for this application, a grid system must be established to define the possible locations where the ship can be positioned at various time steps within a discrete action space. According to the route optimisation environment setting of a similar work applying reinforcement learning for ship route optimisation (Moradi et al., 2022), the grid system in this work is designed with a resolution of 5 km in both longitudinal and latitudinal directions, constructed along the great circle route and perpendicular to it, with 12 points on each side. This grid layout, along with the area of interest, is shown in Fig. 7. The ship being studied operates in the Tyrrhenian Sea, a region characterized by numerous geographical obstacles, such as islands and the mainland, which intersect the great circle route (highlighted in green in Fig. 7). To account for these obstacles, this study utilizes a dataset containing global water depth information from the National Oceanic and Atmospheric Administration (NOAA), specifically the ETOPO1_Bed_c_gmt4.grd dataset (NOAA National Geophysical Data Center, 2009). Since the ship's draft is typically less than 7 m, any location with a water depth of less than 10 m is considered a barrier point. Furthermore, to improve the accuracy of the barrier dataset, small islands that were not represented in the original dataset have been added manually. These setups enable the model to account for navigational safety, including geographical constraints and shallow water conditions.

At each time step, the agent is tasked with determining its deviation from the great circle path. Consequently, the environment is configured such that at each step, the ship advances by one grid column towards its destination along the direction of the great circle, while the agent decides its movement in the direction perpendicular to that. Fig. 8(a) depicts the potential trajectories. In this diagram, the blue dots represent the grid points, while the red dots indicate the permissible positions for discrete observations (as used in the DQN model). This setup means that at each time step, the agent has five potential positions to choose from. Since the distance between adjacent grid points (both in latitude and longitude) is approximately 5 km, the agent operating in a discrete action space moves 5 km in the great circle direction and can choose to move either 0, 5, or 10 km along the direction perpendicular to the great circle.

To take the grid boundary and geographical limitation into consideration, a dynamic action space is implemented in this work. Fig. 8 provides a detailed visualization of the ship's dynamic action space under different conditions. The figure shows several typical scenarios: In scenario **(a)**, the ship operates in an open area with no barriers or grid boundaries, allowing it to freely choose from multiple possible waypoints, represented by red dots. In scenario **(b)**, the ship encounters the grid boundary but no barriers, which

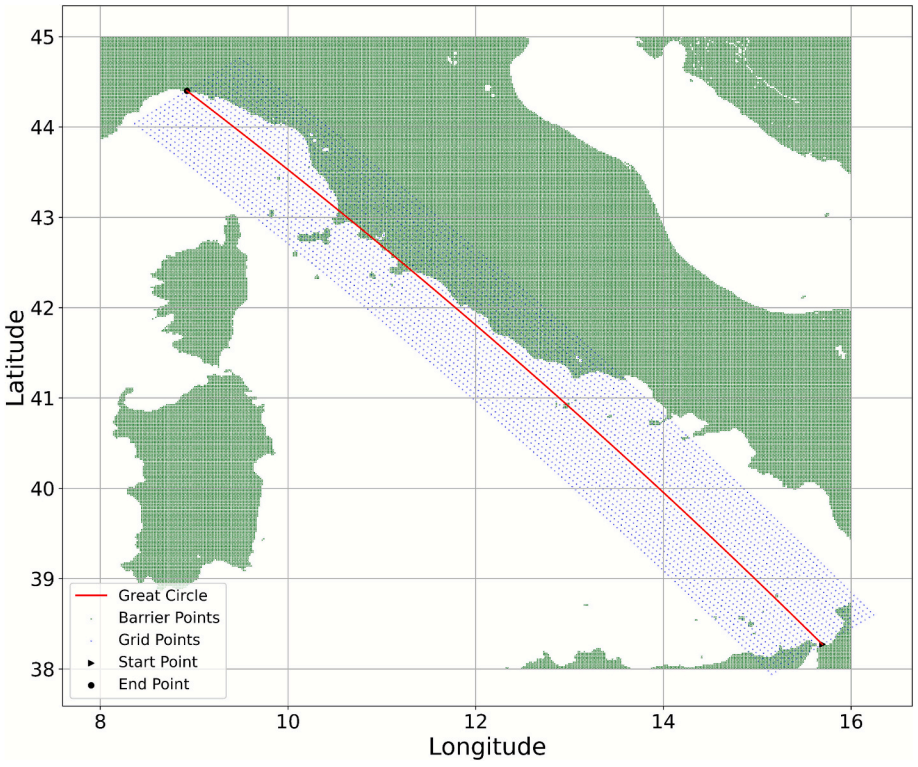


Fig. 7. The grid points and geographical limitation.

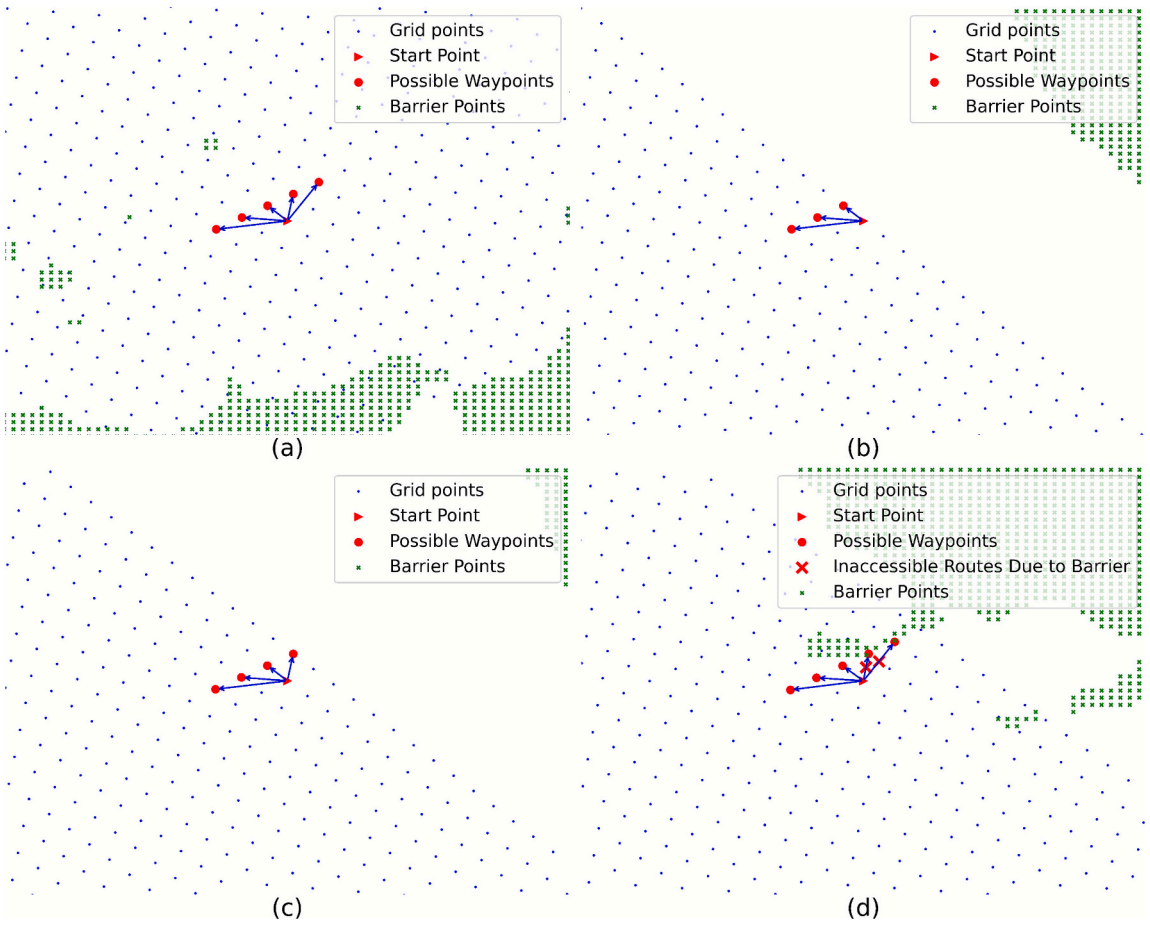


Fig. 8. The dynamic actions regarding the ship's course in four typical conditions; (a) No barrier and boundary close or encountered, (b) Encounter grid boundary but no barrier encountered or close, (c) Close to grid boundary but no barrier and boundary encountered, and (d) Close to barrier but no grid boundary encountered.

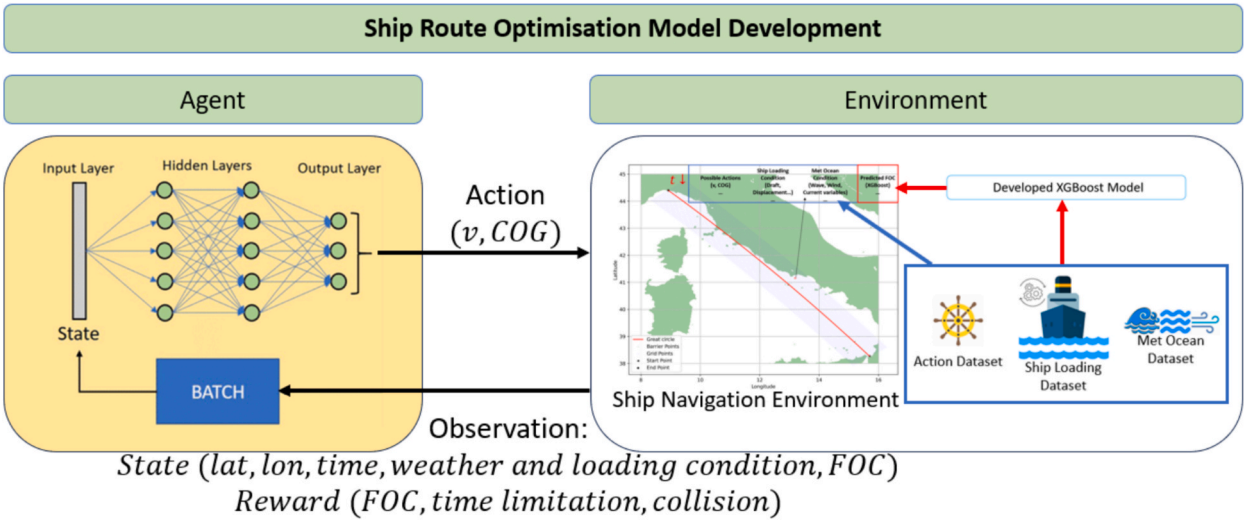


Fig. 9. Framework of the Proposed Ship Route Optimisation Model.

restricts its movement to within the grid limits while still having several possible waypoints. In scenario **(c)**, the ship is near the grid boundary but does not encounter it nor is it close to any barriers, maintaining some flexibility with possible waypoints, though within a constrained space. In scenario **(d)**, the ship is close to barriers, such as land or shallow water, which is indicated by green marks that represent inaccessible barrier points. These barriers are inaccessible and prevent the agent from moving into certain areas, requiring it to alter its course to avoid these obstacles. In addition, during the RL simulation, if the agent gets too close to the barriers, a large penalty will be given.

The agent is also responsible for determining the ship's speed. The speed can be selected from a range of predefined values between 8 and 12 knots based on the ship operating data, in 0.1-knot increments.

2.2.5. Optimisation model development

With the action space, state definitions, and the FOC estimation model already established in **Section 2.2.3 and 2.2.4**, the working principle of DQN-based models illustrated in Fig. 6 could be linked to the ship route optimisation model. Fig. 9 visualises the overall model development procedure of the optimisation framework based on the proposed approach. Before the optimisation process begins, it is necessary to incorporate weather conditions, loading condition, and FOC predictions into the optimisation environment. For each waypoint at each timestamp of a planned voyage, a sub-dataset is generated that includes the weather and loading conditions, the possible actions (variations in SOG and COG) available to the agent, and the corresponding FOC values. These sub-datasets are then combined into a comprehensive dataset containing the relevant information for all grid points across all timestamps, which is subsequently integrated into the navigation environment. At each step, the agent (i.e., the target ship) selects an action consisting of a speed and COG, while the state, which is defined in terms of the ship's geographical location, time, weather and loading condition, and the predicted FOC value, and the corresponding reward related to FOC, time costs, and collision avoidance, are observed from the environment. These elements collectively facilitate the agent's learning process within the optimisation framework.

The development process for the proposed framework is outlined in Table 2, where the interactions between the agent and the ship route optimisation environment are highlighted, along with the integration of the enhanced learning components. Initially, the training process begins with the loading of a predefined agent, denoted as A_0, and the initialization of the environment E_0, which incorporates grid points, barriers, weather and loading conditions, as well as the resulting FOC values. Key outputs, including rewards (R), exploration rates ($\in$), and trajectories (T), are initialized as empty sets to store relevant data throughout the training. During the training loop, for each episode, the environment is reset to its initial state S_0. The agent then iteratively selects actions A_t according to an $\in$-greedy policy. The environment simulation follows, where the action A_t is applied, and subsequent states S_{t+1} and rewards R_{t+1} are observed. Experience is aggregated as n −step transitions $\left(S_t, A_t, R_t^{(n)}, S_{t+n}\right)$ using an n-step buffer, and subsequently stored in the

Table 2
Training procedure of the proposed framework.

1. Initialize Training Process
- $A_0 \leftarrow$ *Load the defined agent, begin with* A_0.
- $E_0 \leftarrow$ *Initialize Environment* E_0, *with grid points, barriers, weather, ship loading and FOC data.*

2. Initialize Key Output Storing
- $R, \in, T \leftarrow \varnothing$: *Initialize empty sets or lists for rewards* R, *epsilon* $\in$, *and trajectories* T.

3. Training Loop (For each episode i in $N_{episodes}$)
- *Reset Environment* :
  - o $S_0 \leftarrow E_0$: *Reset the enviromint, start at initial state* S_0.
- *Episode Execution* :
  - o *While episode is not done* :
    - *Select Action* :
      - *Select Action* A_t *using* $\in -$ *greedy policy from the dueling* $Q-$ *network.*
    - *Simulate Environment* :
      - $S_{t+1}, R_{t+1} \leftarrow E(A_t)$:
        - o *Apply action* A_t *to the environment.*
        - o *Step reward calculated based on* A_t.
        - o *Observe next state* S_{t+1} *and reward* R_{t+1}.
    - *Store Experience* :
      - *Aggregate* $n-$*step transition* $\left(S_t, A_t, R_t^{(n)}, S_{t+n}\right)$ *using* $n-$*step buffer in replay buffer, then store in PER buffer*
- *Replay and learn* :
  - *If PER buffer is sufficient:*
    - o *Sample batch B with priorities & importance weights.*
    - o *Compute targets using Double DQN.*
    - o *Update online dueling Q-network.*
    - o *Update priorities in PER.*
    - o *Periodically update the target network.*
- *Record Results* : *Append total reward* R_i, *final epsilon* $\in_i$, *and trajectory* T_i *for the episode.*

4. Monitor and Log Progress: Supervise the learning process based on the defined logging details.

5. Finalize Training: Return and collect $R, \in, T$ and all recorded metrics.

PER buffer. When sufficient experience has been accumulated, a batch B is sampled from the buffer according to priorities with importance-sampling weights. Learning then proceeds through Double DQN updates, where the online dueling Q-network is optimized, priorities in the PER buffer are updated, and the target network is periodically synchronized. Throughout the episodes, metrics such as cumulative rewards, final exploration rates, and trajectories are recorded to monitor progress. Upon completion of training, the framework collects and returns all recorded outputs, $R, \in$ and T to enable further analysis of the agent's performance.

2.3. Traditional Approach: A* Search-based optimisation

The A* algorithm was implemented to compute optimal routes, using either distance or fuel consumption as the primary cost function. Initially, it optimized solely for distance, which provides a baseline solution for other methods that focus on exploring local optima. To enhance operational efficiency, a variant based on fuel consumption was later introduced. A* is a widely used pathfinding algorithm grounded in the principles of the Bellman equation and optimality theory, which operates using a total cost function as shown in **Eq. (17)**:

$$f(n) = g(n) + h(n) \tag{17}$$

where $g(n)$ is the accumulated fuel cost from the start to the current node, and $h(n)$ is a heuristic estimating the remaining fuel needed to reach the goal. Notably, the FOC values for the A* optimisation process in this work are obtained from the developed XGBoost-based FOC prediction model following the action space defined in **Section 2.2.4**, with a constant service speed for the ship assumed throughout the voyage.

2.4. Intelligent search Algorithm: PSO-based optimisation

PSO is an optimization algorithm inspired by the foraging behaviour of bird flocks introduced by Kennedy and Eberhart (1995). Birds search for food without knowing its exact location but rely on shared and individual experiences to guide their movements. Each bird (or particle) adjusts its direction based on its own best-found position (P_{best}) and the flock's best-known position (G_{best}), gradually converging on the richest food source, which is known as the global optimum.

The PSO algorithm simulates this process through the following steps: Firstly, initializing the positions and velocities of the particle swarm. Secondly, evaluating each particle's fitness (food amount). After that, updating each particle's P_{best} and the global G_{best}. If the termination condition is met, return the best solution; otherwise, repeat the process.

With a population size of N in a D-dimensional space assumed, the position and velocity updating process is given by:

$$\begin{cases} V_{i,j}(t+1) = \omega_{PSO}(t) \times V_{i,j} + c_{1,PSO} \times rand_1 \left[P_{best_{i,j}} - x_{i,j}(t)\right] + c_{2,PSO} \times rand_2 \times \left[G_{best} - x_{i,j}(t)\right] \\ x_{i,j}(t+1) = x_{i,j}(t) + V_{i,j}(t)(t+1) \end{cases} \tag{18}$$

where $V_{i,j}(t)$ and $X_{i,j}(t)$ represent the j-th dimensional velocity and position components of the i-th particle at the iteration t. $i = 1, 2, 3, \cdots, N$ represents the index of particle, while N is the population size, $j = 1, 2, 3, \cdots, D$ denotes the index of dimension. $c_{1,PSO}$ and $c_{2,PSO}$ are the acceleration coefficients. $rand_1$ and $rand_2$ are random numbers unevenly distributed between [0,1]. ω_{PSO} is the inertia weight which regulate the inertia of particle velocity.

In this work, both A* (a typical traditional deterministic algorithm) and PSO (a typical intelligent searching algorithm) were applied to optimise ship routing under identical environmental conditions. These methods have been applied to provide baseline comparisons and validate the performance insights of the proposed RL-based approach.

3. Result and Discussion

In this section, the results obtained from the stages outlined previously are discussed. In particular, there is a focus on the performance of the FOC model during both the validation and test phases. Additionally, details of the setup and outcomes of the route optimisation process, utilising the proposed DQN approaches, are presented. The computer resources and the related software specifications in this work are shown in Table 3.

3.1. Xgboost FOC model

The datasets, following the comprehensive filtering and cleaning process introduced in **Section 2.1**, consist of 19,149 min-based

Table 3
The applied hardware, software and packages in this study.

Resources	Specifications
Processor	CPU: 11th Gen Intel(R) Core(TM) i7-11800H @ 2.30 GHzGPU: NVIDIA GeForce RTX 3070 Laptop GPU
Memory	16 GB
Programming language	Python
Machine learning and Optimisation Algorithm packages	scikit-learn, gym, queue, pyswarms

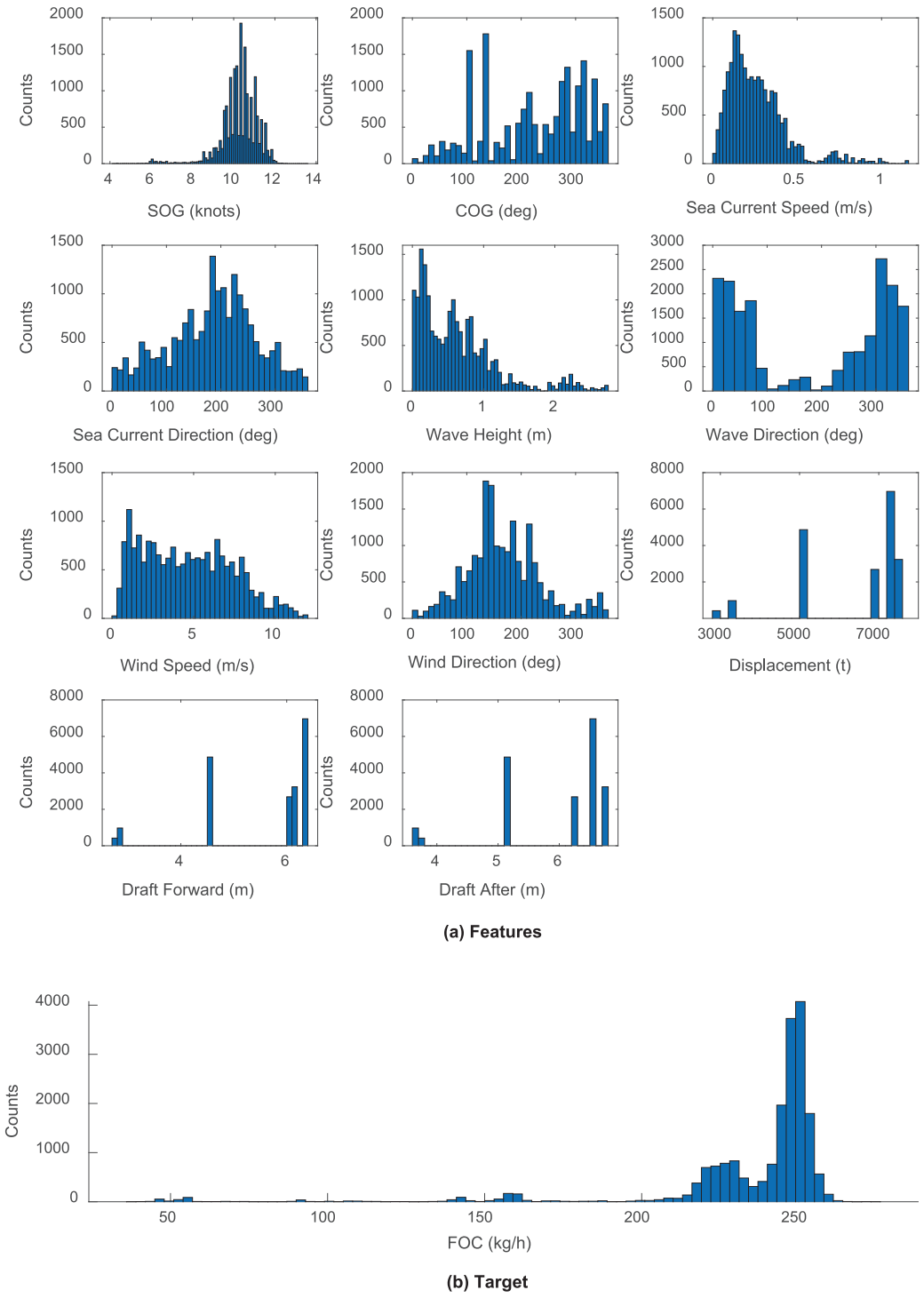


Fig. 10. Data distribution for FOC model development and evaluation: **(a)** Features; **(b)** Target.

samples specifically curated for the development of the FOC model. These datasets will be the basis for model development and subsequent analysis. Fig. 10 presents histogram plots to provide insights into the data's distribution and key characteristics of the features and target. In these plots, the vertical axis indicates the number of data points within each histogram bin, which offers a clear visualization of the frequency distribution across various ranges. To further describe the data used in this work, statistical analysis (Table 4) has been conducted for each feature and the target (Xie at al., 2023).

The XGBoost FOC model was fine-tuned using a comprehensive hyperparameter search strategy to ensure optimal performance. As shown in Table 5, the hyperparameters explored included the number of estimators (n_estimators), which was varied from 500 to 1500 in increments of 100; the maximum depth of each decision tree (max_depth), which ranged from 2 to 10 with a step size of 1; and the learning rate (learning_rate), which was fine-tuned between 0.005 and 0.01 with a step size of 0.001. Based on recent research on machine learning models for ship fuel consumption and energy prediction (Zhang et al., 2024; Hu et al., 2025), a 5-fold cross-validation strategy was employed. Through the 5-fold cross-validation process introduced in **Section 2.1.6**, the optimal hyperparameters were determined to be 800 estimators, a maximum depth of 9, and a learning rate of 0.01. These settings were selected based on their ability to minimize the prediction error on the validation set, which yields a mean MAE of 2.02 kg/h. On the unseen dataset, the XGBoost model exhibited excellent generalization capabilities, achieving an MAE of 1.94 kg/h. Additionally, the model recorded a MAPE of 0.89 %, and a high coefficient of determination ($R^2 = 0.99$). This good performance is consistent with similar published works, such as R2 = 0.989 reported by Moradi et al. (2022) and R2 = 0.9958 by Xie et al. (2023) and has further confirmed the model's effectiveness. Its performance on the unseen test dataset is visualized in Fig. 11. The scatter plot provided in the figure further illustrates the model's predictive accuracy, where the predicted FOC values are plotted against the true FOC values for the test dataset. The points are tightly clustered around the diagonal line, which represents perfect predictions. This tight clustering indicates that the predicted values closely match the actual FOC values across the entire range of data. The alignment along the line also reflects the model's strong linear correlation between predicted and actual values. This high level of accuracy on the test dataset indicates that the FOC model is effective for predicting FOC for unseen voyages.

3.2. Route optimisation models

In this study, a segment of the route, measuring 913.8 km, from the Strait of Messina to Genoa in the Tyrrhenian Sea, is selected as a case study to evaluate the route optimisation approach. The original path taken by the ship is illustrated in Fig. 12, with the journey commencing at 03:12:00 on May 27, 2023. During this voyage, the key loading parameters of ship displacement, draft forward, and draft aft were recorded as 7600 tonnes, 6.2 m, and 6.6 m, respectively. Dynamic weather data for the entire area covered by the grid points was obtained from the CMEMS databases. This data retrieval was based on the Nearest Neighbour Method described in **Section 2.1.4**, which applies both time stamps and geographic locations to ensure precise alignment with the ship's position over time. The weather data, which includes information on waves, currents, and wind, is dynamic and varies throughout the journey. This dynamic nature of the weather is illustrated in Figs. A1-A3 in the **Appendix 1**, which show changes in wave height, current strength, and wind speed over time for the investigated area. Given the action space defined by the ship's speed and course, along with the recorded loading conditions and the dynamic weather information, the XGBoost model can be applied to estimate FOC values for voyage environment setups.

As discussed in **Section 2.2**, both standard DQN and its enhanced version have been developed to evaluate their effectiveness in optimising ship routes under varying conditions. Two distinct scenarios are designed to achieve different optimisation objectives. In the first scenario, the primary goal is to minimize fuel consumption without imposing any time penalty on the journey. In the second scenario, the optimisation is dual-objective: fuel consumption is still minimized, but a time penalty term is introduced. The reward function of these scenarios can be presented as:

Table 4
Descriptive Statistics of FOC Model Training Dataset.

	Min	Mean	Max	25 %	50 %	75 %	Std
SOG (kn)	4.19	10.27	13.54	9.88	10.30	10.90	0.91
COG (deg)	0.91	223.74	359.00	137.14	242.72	307.78	91.24
Sea Current Speed (m/s)	0.00	0.26	1.14	0.13	0.22	0.34	0.18
Sea Current Direction (deg)	2.40	184.93	359.14	133.78	190.94	238.61	79.97
Wave Height (m)	0.01	0.63	2.75	0.19	0.52	0.86	0.55
Wave Direction (deg)	0.05	181.86	359.93	42.41	233.77	309.88	134.26
Wind Speed (m/s)	0.18	4.63	11.87	2.21	4.43	6.74	2.73
Wind Direction (deg)	0.00	167.14	359.05	124.67	158.55	211.34	68.33
Displacement (t)	2900.00	6427.15	7451.00	5100.00	7395.00	7395.00	1337.49
Draft Forward (m)	2.70	5.56	6.40	4.50	6.13	6.40	1.08
Draft After (m)	3.60	5.97	6.78	5.10	6.55	6.55	0.91
FOC (kg/h)	38.88	236.23	275.31	230.64	246.69	250.29	30.49

Table 5
Model hyperparameters tuning, validation and test results.

Model	Hyperparameterstuned	Optimal Hyperparameters and Validation Metrics	Performance Metrics on Test Dataset
XG	n_estimators ∈ [500, 1500, step:100], max_depth ∈ [2, 10, step:1], learning_rate ∈ [0.005, 0.01, step: 0.001]	'n_estimators': 800, 'max_depth ': 9, 'learning_rate': 0.01 **MAE = 2.02 kg/h**	MAE = 1.94 kg/h MAPE = 0.89 % R^2 = 0.99

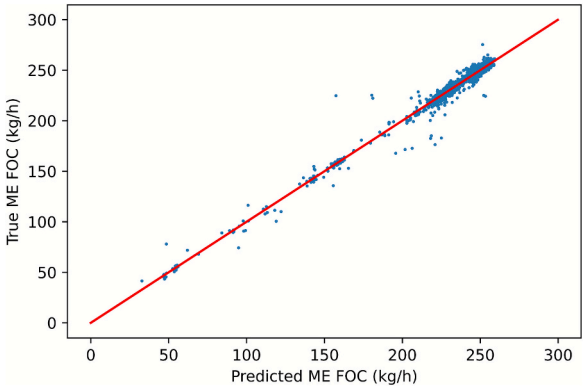


Fig. 11. XGBoost model performance: true value vs. predicted value over test dataset.

Fig. 12. Original route taken by the ship.

$$R_{Total} = \begin{cases} \sum_{t=1}^{T_{step}} \left(- FOC_{hr}^{t} \bullet t_{step}^{t} - \omega_{invalid\_move} \bullet \delta_{invalid\_move}^{t} \right), \\ \qquad\qquad Scenario1 \\ \sum_{t=1}^{T_{step}} \left[- FOC_{hr}^{t} \bullet t_{step}^{t} - \omega_{time} \bullet \left(t_{step}^{t} - t_{max} \right) - \omega_{invalid\_move} \bullet \delta_{invalid\_move}^{t} \right], \\ \qquad\qquad Scenario2 \end{cases} \tag{19}$$

where

R_{Total}: Total accumulated reward over the entire episode.

T_{step}: Total number of steps in the episode.

FOC_{hr}^{t}: FOC in kg/h at time step t.

t_{step}^{t}: Travel time duration for the action taken at step t.

$\delta_{invalid\_move}^{t}$: Indicator variable, 1 if the move at step t is invalid (e.g., enters restricted areas, crashes into islands or geographic barriers, or violates grid boundaries), and the episode terminates immediately when entering the restricted areas or violating grid boundaries; otherwise, 0.

$\omega_{invalid\_move}$: Penalty weight for an invalid move.

ω_{time}: Penalty weight for deviation from target step time, penalizing longer than expected step durations.

t_{max}: Target (maximum acceptable) step time in hours.

In this work, a scalarisation strategy is adopted in the RL-based models to address the multi-objective nature of ship routing and speed optimization. Specifically, fuel consumption and time considerations are combined into a single scalar reward using a weighted sum formulation, while invalid actions (entering the restricted areas or violating grid boundaries) are enforced as hard constraints by

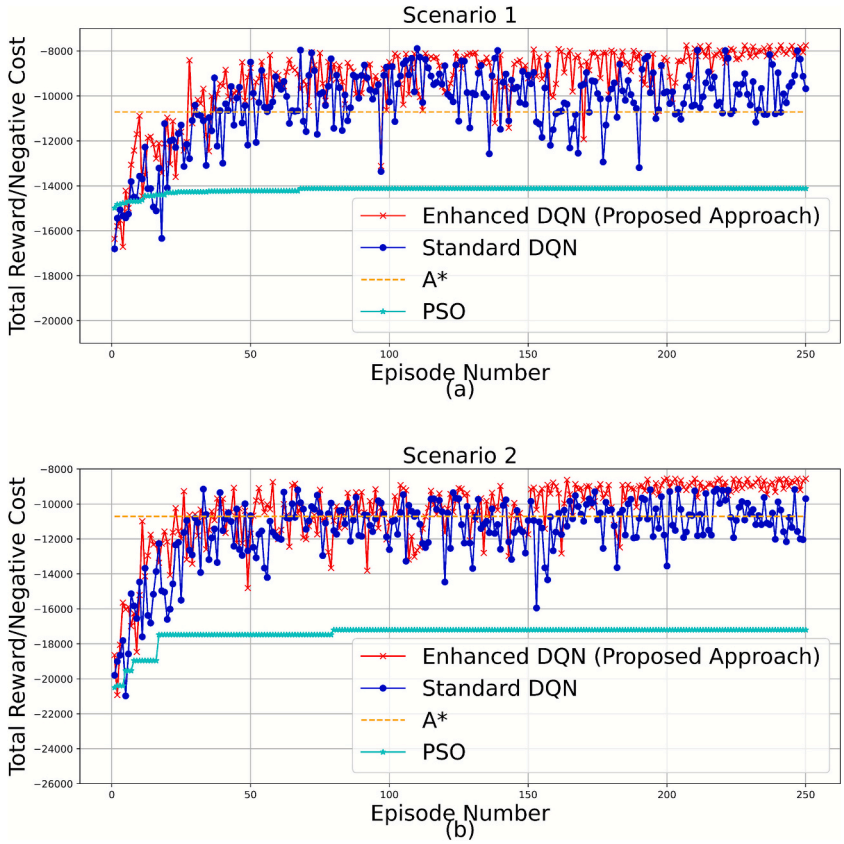


Fig. 13. Reward/Negative Cost Over Training Steps for the Investigated Approaches; (a) Scenario 1 and (b) Scenario 2.

terminating the episode immediately upon violation with a large penalty. Thus, the formulation represents a multi-objective RL framework with environmental constraints, where actions that violate safety are discouraged through penalties and termination.

In the context of PSO, the models are typically optimized using a fitness function, which is conceptually a cost function rather than a reward. To align with this paradigm, in both scenarios described, the negative value of the total reward ($-R_{Total}$) has been applied as the cost function of PSO.

In terms of A* Search, the setup is the same in Scenario 1 and 2, with the cost function of A* Search defined by **Eq. (17)**. Due to the limitation of implementing the penalty term induced by invalid move to the cost function, grid points labelled as '*invalid_move*' were removed from the A* navigation environment to prevent invalid actions (e.g., enters restricted areas, crashes into islands or geographic barriers, or violates grid boundaries).

The ship requires approximately 49.34 h to reach its destination at a service speed of 10 knots when following its original route. Therefore, for Scenario 2, $t_{max} = 49.34/177$ is set for each step to encourage the agent to explore lower FOC options while accounting for the associated time cost. Notably, the study does not consider safety constraints related to wave height since wave heights are generally below 1 m (as illustrated in Fig. A1).

For model training, the optimisation models were trained using the fine-tuned hyperparameters outlined in Table A1, Table A2 and Table A3, provided in the Appendix. The subsequent content provides a detailed analysis of the optimisation outcomes after training. Fig. 13 illustrates the reward history of both the standard DQN and the enhanced DQN during the learning process for the two different scenarios given exactly the same episodes, which provides insights into their respective performances over the course of the training episodes.

In Scenario 1 (Fig. 13(a)), where the objective is solely to minimize fuel consumption without considering the time cost, the reward history shows a marked improvement in total rewards as training progresses. The enhanced DQN consistently achieves higher total rewards compared to the standard DQN, especially after the initial episodes. This indicates that the enhanced DQN is more effective in learning optimal policies that result in lower fuel consumption. The standard DQN demonstrates a more fluctuating learning curve, with rewards exhibiting greater variability, particularly in the early episodes. As training continues, the variability decreases, and the rewards begin to stabilize. However, the enhanced DQN shows a more stable convergence, with less fluctuation in reward values and a quicker attainment of higher total rewards. In Scenario 2 (Fig. 13(b)), where the optimisation objective includes both minimizing fuel consumption and time costs, the reward trends display similar patterns of convergence for both algorithms. Both the standard and enhanced DQNs show initial episodes with low total rewards, which indicates the exploration phase where suboptimal actions are more frequent given the large size of observation space. However, as training progresses, the enhanced DQN again outperforms the standard DQN, consistently achieving higher rewards throughout the episodes, which in turn indicates that the overestimation in the standard DQN typically leads to instability, slower convergence, or suboptimal performance. The more volatile behaviour of the Standard DQN in both scenarios could be indicative of overestimation-related issues. This clearly illustrates the improvement in the enhanced version, specifically, the application of Double DQN in terms of mitigating overestimation. The reward histories suggest that the enhanced DQN demonstrates superior learning efficiency and stability in both scenarios and also achieves higher rewards. This improvement in performance can be attributed to the modifications and enhancements (discussed in **Section 2.2.2**) integrated into the DQN framework, which contribute to a better exploration–exploitation balance and more effective learning of the complex decision-making policies required for optimal route planning in maritime environments.

Comparative results with PSO and A* further highlight the advantages of the proposed approach. The A* algorithm, being a

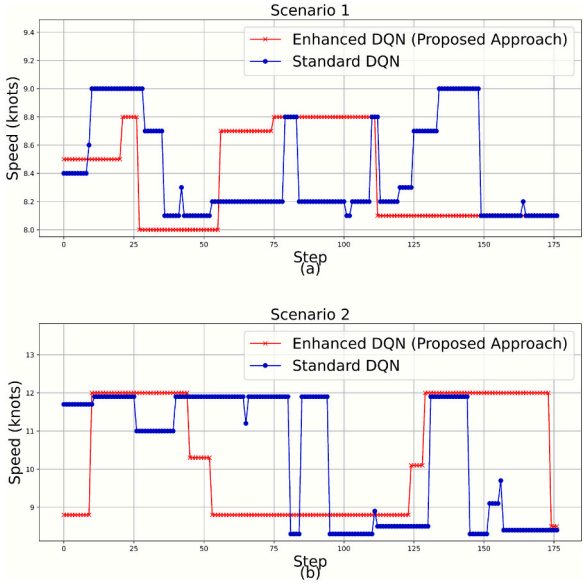


Fig. 14. Optimal Speed Profiles from Enhanced and Standard DQN Approaches; (a) Scenario 1 and (b) Scenario 2.

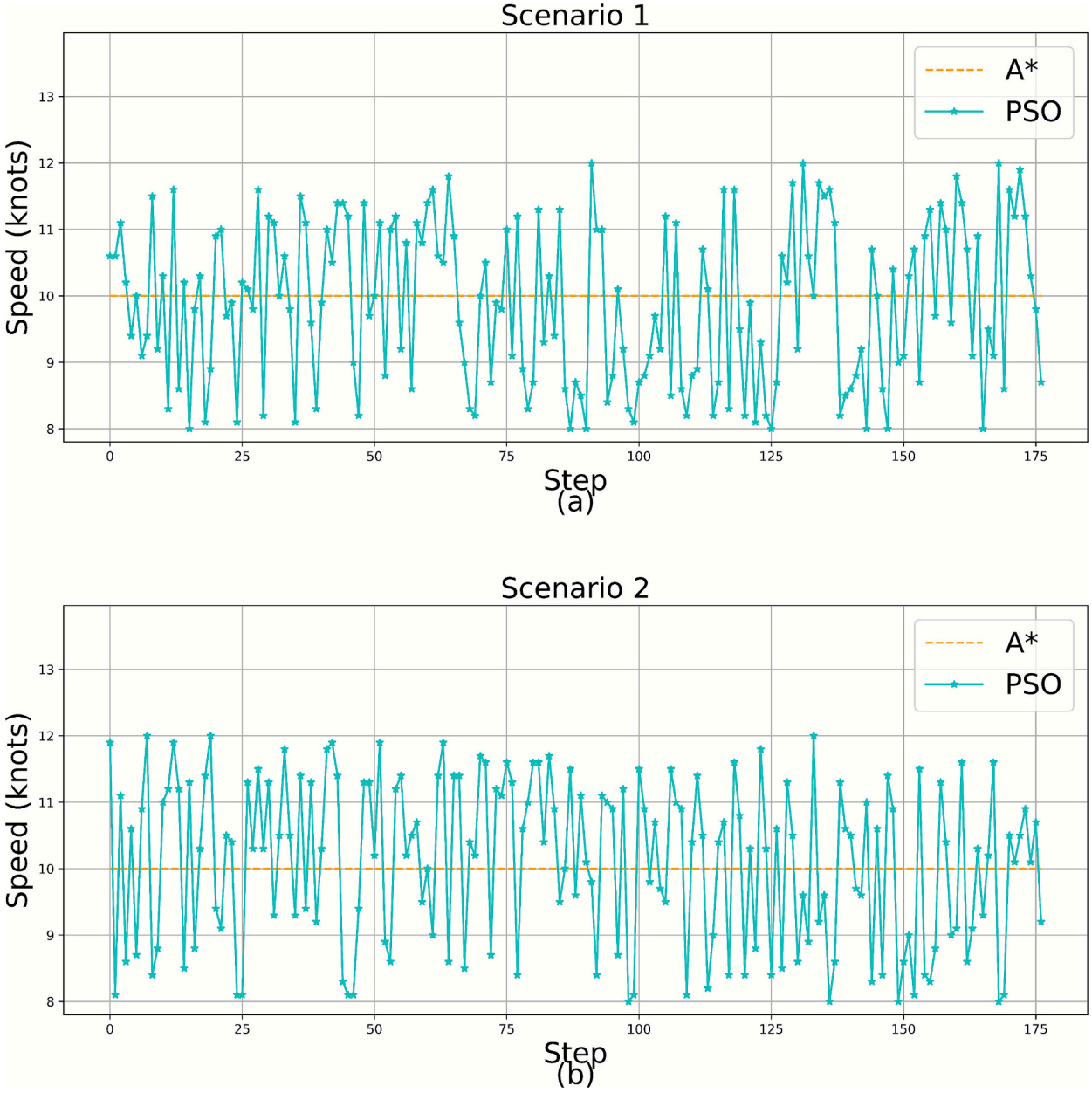


Fig. 15. Optimal Speed Profiles from A* Search and PSO Approaches; (a) Scenario 1 and (b) Scenario 2.

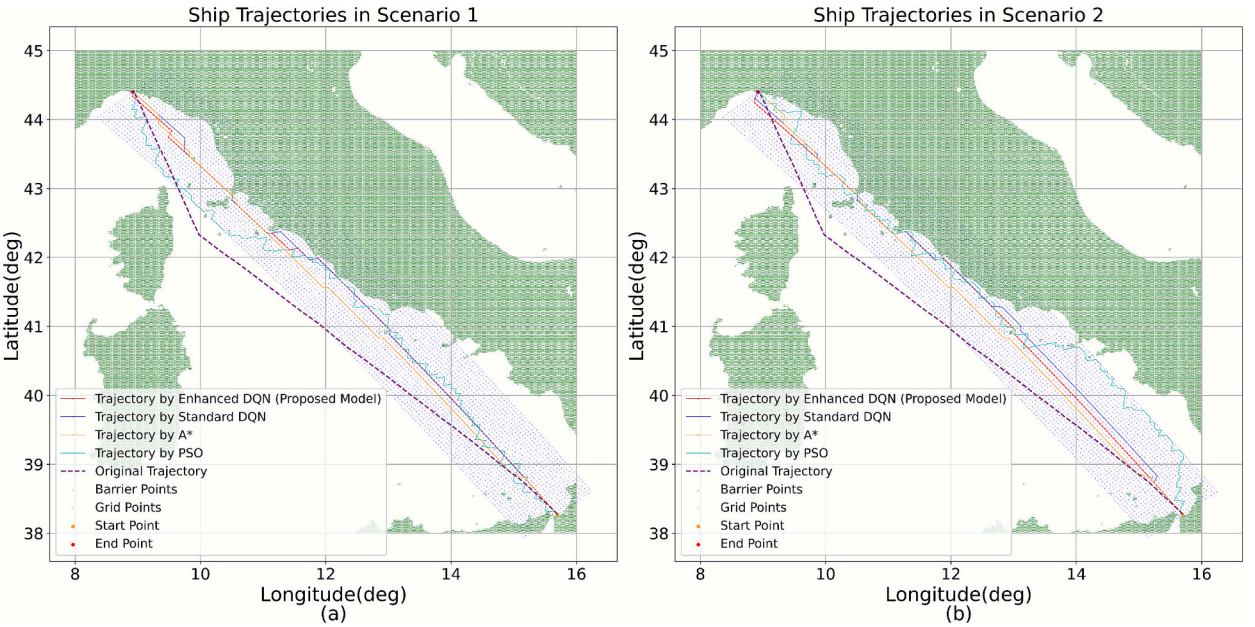


Fig. 16. Optimal Ship Trajectories from the Investigated Approaches; (a) Scenario 1 and (b) Scenario 2.

Table 6
Comparison of optimisation algorithms over the baselines.

Models and Baselines	**Scenario**	**Computation Time (s)**	**Time** Voyage Duration (h)	Compare to Baselines (%)	**FOC** FOC (kg)	Compare to Baselines (%)
Enhanced DQN (Proposed approach)	1	10,312	58.86	+19.29 over **Baseline 1** and **2**	7754.36	−29.01 over **Baseline 1**, −34.54 over **Baseline 2**
	2	10,645	49.30	−0.08 over **Baseline 1** and **2**	8551.13	−21.72 over **Baseline 1**, −27.81 over **Baseline 2**
Standard DQN	1	8083	59.15	+19.88 over **Baseline 1** and **2**	7936.30	−27.35 over **Baseline 1**, –33.00 over **Baseline 2**
	2	8608	49.32	−0.04 over **Baseline 1** and **2**	9140.23	−16.33 over **Baseline 1**, –22.84 over **Baseline 2**
PSO	1	9691	73.42	+48.80 over **Baseline 1** and **2**	14117.89	+29.24 over **Baseline 1**, +19.18 over **Baseline 2**
	2	9929	70.90	+43.70 over **Baseline 1** and **2**	14501.79	+32.76 over **Baseline 1**, +22.42 over **Baseline 2**
A*	/	94	48.68	−1.34 over **Baseline 1** and **2**	10711.94	−1.94 over **Baseline 1**, −9.57 over **Baseline 2**
Baseline 1: Original Route After Retrofit at 10 knots	/	/	49.34	/	10923.59	−7.79 over **Baseline 2**
Baseline 2: Original Route Before Retrofit at 10 knots	/	/	49.34	/	11846.15	+8.45 over **Baseline 1**

deterministic path planning method that does not involve episodic learning, is represented as a horizontal reference line indicating its negative constant total cost. Since the time penalty terms are incorporated into the reward/cost functions of the standard DQN, enhanced DQN, and PSO, it is more appropriate to compare the A* results with these methods in Scenario 1, where time cost is not considered. While A* outperforms PSO, it is clearly outmatched by both the standard and enhanced DQN approaches. The enhanced DQN, in particular, surpasses A* significantly after the early training episodes. This illustrates the advantage of reinforcement learning-based methods, especially the enhanced DQN, in exploring and exploiting more efficient routing strategies beyond the static solution provided by A*. Even when incorporating the time penalty term in Scenario 2, the enhanced DQN still achieves a higher overall reward than A* after convergence.

In both Scenario 1 and Scenario 2, the performance of PSO is notably inferior compared to the learning-based approaches and the deterministic A* algorithm. As shown in the figures, PSO converges early in the training episodes to a suboptimal reward level and maintains a nearly flat curve, thereafter, indicating a lack of further exploration. This behaviour suggests that PSO may be prone to getting trapped in local optima, particularly in high-dimensional and constrained optimization spaces (Yao et al., 2024), which is especially relevant given the complex maritime environment described in this work.

Fig. 14 presents the optimal speed profiles generated by the standard DQN and the enhanced DQN for both Scenario 1 and Scenario 2. In Scenario 1 (Fig. 14(a)), the two approaches exhibit noticeably different behaviours. The standard DQN demonstrates higher variability, which frequently oscillates between different speed settings. These abrupt transitions reflect a more dynamic decision-making process, which may lead to suboptimal fuel efficiency due to constant speed adjustments. In contrast, the enhanced DQN maintains a more stable and consistent speed profile, primarily operating within a range of 8.0 to 8.8 knots. This suggests a more refined and deliberate optimization strategy focused on minimizing fuel consumption through steady-state operation. In Scenario 2 (Fig. 14(b)), the standard DQN continues to exhibit more abrupt fluctuations, likely in an attempt to reconcile the dual objectives. However, this results in an unstable profile, in contrast, the enhanced DQN displays a more balanced and strategic response with fewer abrupt transitions. The smoother progression reflects a more effective learning strategy that manages the trade-off between time and fuel consumption.

Fig. 15 presents the speed profiles of A* and PSO in both scenarios. A* maintains a constant service speed of 10 knots as predefined. PSO, however, shows erratic fluctuations between 8 and 12 knots, which reflects poor convergence and a likely tendency to fall into local optima. This unstable behaviour corresponds to its weak performance in Fig. 13(a) in terms of convergence. In Scenario 2, with the added time penalty, PSO reacts by selecting higher speeds more frequently (up to 12 knots), but its speed profile remains chaotic and unstructured. While the time penalty encourages faster speeds, PSO fails to balance this with fuel efficiency.

Fig. 16 illustrates the optimal ship trajectories generated by the enhanced DQN, standard DQN, A*, and PSO in Scenario 1 (Fig. 16(a)) and Scenario 2 (Fig. 16(b)). Across both scenarios, all models successfully avoid collisions with the defined barrier zones, which indicates that the trajectory planning mechanisms in each approach correctly integrate environmental constraints. In Scenario 1, the enhanced DQN produces a smooth and efficient trajectory that strategically deviates from the original path to bypass barriers with minimal excess distance. The standard DQN also avoids obstacles but generates a more irregular path. The A* algorithm yields a relatively direct route that prioritizes minimal-FOC navigation while avoiding barriers. In contrast, the PSO trajectory is notably fluctuating and fragmented. In Scenario 2, the enhanced DQN adapts to the time penalty term by producing a slightly more direct route. The standard DQN continues to show a jagged path at some periods. The A* trajectory remains identical to that in Scenario 1 as predefined. The PSO path again appears inconsistent and disorganized. These erratic paths, in conjunction with its poor speed consistency (Fig. 15) and low total rewards (Fig. 13), further supports the observation that PSO is prone to falling into local optima and lacks the refinement needed for globally effective decision-making when applied to the established ship environment in this work.

To quantify the savings attributable to the proposed route and operational optimisation method, as well as the savings from the installation of the GR system, two baselines have been established. The first baseline (Baseline 1) is defined by maintaining the ship at a constant service speed of 10 knots along its original route. For this baseline, the developed FOC model, based on post-retrofit data, is applied to estimate the FOC values along the route under the same loading conditions (Displacement, draft forward/after) set up for the RL simulation. The time required to complete the journey and the total FOC derived from this model are recorded as Baseline 1 metrics. Notably, the ship was retrofitted with the GR system in early May 2023, and the FOC model used within the RL environment was developed using data collected after the installation of this system.

Additionally, a dataset representing the ship's performance prior to the installation of the GR system was also available. This pre-retrofit dataset was utilized to develop a separate FOC model to characterize the ship's fuel efficiency before the GR system installation. This model development followed the same methodological pipeline as described in **Section 2.1**. The FOC model developed for pre-retrofit conditions demonstrated a high coefficient of determination R^2 of 0.994 over the unseen test dataset, which confirms its effectiveness in estimating FOC for voyages conducted before the retrofit. Given this pre-retrofit FOC model, the FOC can be estimated along the original route under the same loading conditions as defined for Baseline 1. The corresponding time and total FOC from this estimation constitute Baseline 2. As shown in Table 6, Baseline 1 estimated an FOC of 10923 kg over 49.34 h, which is 7.79 % lower

Table 7
Comparison of the optimisation results with/without FOC prediction model.

Model	Scenario	FOC Prediction Model?	Voyage Dura ion (h)	FOC (kg)
Enhanced DQN	2	Yes	49.30	8551.13
	2	No	41.23	12010.76

than Baseline 2's 11846 kg over the same duration. This confirms the saving achieved by the GR system over the target journey.

Table 6 presents a comparative analysis of the enhanced DQN, standard DQN, PSO, and A* algorithms across two scenarios, evaluated against two baseline strategies.

In Scenario 1, the enhanced DQN achieves a voyage duration of 58.86 h, which is 19.29 % longer than both Baseline 1 and Baseline 2. Despite the increased duration, it significantly reduces FOC to 7754.36 kg, with a reduction of 29.01 % compared to Baseline 1 and 34.54 % compared to Baseline 2. The standard DQN also achieves fuel savings with an FOC of 7936.30 kg, with a reduction of 27.35 % and 33.00 % over Baseline 1 and 2, respectively, at a longer voyage duration of 59.15 h (+19.88 %). The PSO approach achieves a much longer voyage duration of 73.42 h (+48.80 %) and significantly higher FOC at 14117.89 kg, which is 29.24 % and 19.18 % higher than Baseline 1 and Baseline 2, respectively. This again suggests that PSO converges to suboptimal solutions and fails to balance fuel efficiency and route planning effectively with the computation resources and ship environment established in this work. The A* algorithm, which operates deterministically at a fixed speed, completes the route in 48.68 h. This indicates the voyage duration slightly faster than the 49.34-hour baselines, but only reduces FOC to 10711.94 kg (a 1.94 % and 9.57 % reduction compared to Baseline 1 and 2, respectively).

In Scenario 2, the enhanced DQN effectively adapts by achieving a voyage duration of 49.30 h, slightly undercutting the 49.34-hour threshold (−0.08 %) while still reducing FOC to 8551.13 kg, with a 21.72 % and 27.81 % reduction over the two baselines. The standard DQN also satisfies the time requirement with 49.32 h (−0.04 %) and achieves FOC reductions of 16.33 % and 22.84 %. In contrast, PSO again shows inefficiency in Scenario 2, with a long voyage duration of 70.90 h (+43.70 %) and a high FOC of 14501.79 kg, which is 32.76 % and 22.42 % higher than the two baselines. The A* configuration remains consistent with Scenario 1, and its performance metrics have already been compared above.

Computation time further highlights the practical trade-offs between the evaluated approaches. While A* is extremely fast, which requires only 94 s, it offers only modest FOC savings and lacks the adaptability needed for multi-objective optimization. In contrast, both the standard DQN and enhanced DQN require higher computation times (approximately 8,000–11,000 s) but deliver significantly better overall performance, particularly the enhanced DQN, which achieves superior fuel efficiency while considering the time cost. PSO, despite having a computation time comparable to the enhanced DQN, fails to justify its resource consumption according to its poor optimization outcomes. While the performance of PSO can often be improved by increasing the number of particles, it has been found that this comes at the cost of significantly greater computation time and resource demands (Engelbrecht, 2007). Given identical hardware and software conditions, as well as the comparable runtime to the enhanced DQN, PSO failed to find an optimal solution when given the ship navigation environment established in this work.

Furthermore, to evaluate the contribution of the XGBoost-based FOC prediction model to building the shipping environment of the route and speed optimisation framework, an ablation study has been conducted by training the enhanced DQN model with and without the XGBoost-based FOC predictor. In the comparative setup, the FOC penalty term $-FOC_{hr}^{t} \bullet t_{step}^{t}$ from **Eq. (19)** in Scenario 2 was removed to isolate the impact of fuel consumption guidance on the optimisation outcome. In the absence of the FOC penalty, the agent consistently selected higher speeds (mostly around 12 knots) to minimise travel time, which led to a shorter voyage duration. However, this resulted in the cost of substantially increased fuel consumption. The results are summarised in Table 7. These results demonstrate the critical role the FOC model plays in balancing the competing objectives of fuel efficiency and travel time. While the model without FOC information optimises for the time cost, the inclusion of the FOC model enables a more nuanced trade-off, which contributes to a more fuel-efficient routing decision.

The reduction in FOC would lead to a corresponding decrease in CO_2 emissions when employing the proposed optimisation models. The general cargo vessel in this study is powered by Heavy Fuel Oil (HFO), and thus a Carbon Emission Coefficient (CEC) of 3.15 is applied to assess CO_2 emissions and the reduction achieved using the proposed route and speed optimisation models (Chen et al., 2024). Table 8 presents the CO_2 emissions achieved by the proposed approach and the associated reductions relative to the baselines. The proposed enhanced DQN model can achieve a reduction of 9983 kg and 12889 kg of CO_2 in Scenario 1 compared to Baseline 1 (34409 kg) and Baseline 2 (37315 kg), respectively. When accounting for time limitations, the proposed model can achieve reductions of 7473 kg and 10379 kg of CO_2 in Scenario 2 compared to Baseline 1 and Baseline 2, respectively. Overall, these results demonstrate the effectiveness of the proposed optimisation models in significantly reducing both fuel oil consumption and in turn the associated CO_2 emissions, which potentially contribute to more sustainable and environmentally friendly ship operations.

Table 8
Result of CO_2 emission reduction achieved by the proposed approach compared to the baselines.

Models and Baselines	Scenario	CO_2 Emissions CO_2 (kg)	Reduction over Baselines
Enhanced DQN (Proposed approach)	1	24,426	9983 kg (29.01 %) over **Baseline 1**, 12889 kg (34.54 %) over **Baseline 2**
	2	26,936	7473 kg (21.72 %) over **Baseline 1**, 10379 kg (27.81 %) over **Baseline 2**
Baseline 1: Original Route After Retrofit at 10 knots	/	34,409	2906 kg (7.79 %) over **Baseline 2**
Baseline 2: Original Route Before Retrofit at 10 knots	/	37,315	/

4. Conclusion

This study proposes a novel methodology for route optimisation using the XGBoost model and enhanced DQN to optimise the operational efficiency of a general cargo ship navigating the Tyrrhenian Sea. Specifically, a generic ship model has been developed using XGBoost to accurately predict FOC under varying operational and environmental conditions. The predicted FOC values are then integrated into the RL environment of an enhanced DQN, which incorporates advanced learning techniques such as Prioritized Experience Replay, Double Q-Learning, Dueling Networks, and Multi-Step Learning. These techniques significantly enhance the learning efficiency and stability compared to the standard DQN. The proposed approach is applied to optimise ship operations by considering multiple factors, including FOC, safety considerations, voyage time, and geographical limitations. This comprehensive optimisation strategy facilitates more effective decision-making for route planning, ultimately reducing fuel consumption and operational costs while ensuring compliance with navigational and safety constraints. The main findings of the research are as follows:

i. The FOC model developed using XGBoost demonstrated excellent predictive accuracy, achieving an MAE of 1.94 kg/h, a MAPE of 0.89 %, and an R^2 of 0.99 on an unseen test dataset. These results confirm the model's superior performance in modelling and predicting fuel consumption for the target application.
ii. The enhanced DQN demonstrated superior performance over the standard DQN and PSO during the training process with higher total rewards/costs. Although PSO shows a faster and more stable convergence, it is prone to falling into local optima and end up with poor optimisation outcomes. In addition, the speed profiles selected by the enhanced DQN were more stable and consistent than those produced by the standard DQN and PSO. This indicates that the enhanced DQN is more effective and stable in learning optimal policies that result in reduced fuel consumption in a complex environment with time cost and geographical limitations considered.
iii. The enhanced DQN achieved significant FOC reductions compared to the standard DQN, A*, PSO, and baseline routes. In Scenario 1, it reduced FOC by up to 34.54 % over the pre-retrofit baseline, outperforming the standard DQN and substantially exceeding the modest savings of A*, as well as the high fuel consumption of PSO. In Scenario 2, it maintained strong performance with up to 27.81 % fuel savings over the pre-retrofit baseline, while considering time cost.
iv. The installation of the GR system contributed to additional fuel savings beyond those achieved through route optimisation alone. By comparing Baseline 1 and Baseline 2, it was observed that the GR system itself resulted in a 7.79 % reduction in FOC over the target journey.
v. The proposed model achieves significant CO_2 reductions, with a decrease of 9983 kg (29.01 %) and 12889 kg (34.54 %) in Scenario 1 compared to Baseline 1 and Baseline 2, respectively, and 7473 kg (21.72 %) and 10379 kg (27.81 %) in Scenario 2 over the same baselines when accounting for time costs.

Overall, the proposed approach offers a comprehensive and robust solution for maritime route planning, capable of significantly reducing FOC and CO_2 emissions while adhering to navigational and safety constraints. Additionally, the non-graph-based and self-learning nature of RL facilitates its integration into future digital twin-driven autonomous solutions more effectively than traditional approaches. However, some challenges remain. For example, the FOC model could potentially be further improved by incorporating additional oceanographic variables during the model development stage, such as swell height and direction, or sea surface temperature (Li et al., 2022; Du et al., 2022). Moreover, the current study does not account for maritime traffic in the investigated area, which can impact route efficiency and safety. As a next step, incorporating ship congestion and traffic patterns into the reinforcement learning environment is planned to enable more comprehensive optimisation in real-world maritime conditions. In this context, heavy traffic areas will be represented as dynamic barriers, integrated alongside static barriers such as islands and other geographical constraints, to guide the training of the optimisation framework in a more realistic shipping environment.

CRediT authorship contribution statement

Yi Zhou: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Serkan Turkmen:** Writing – review & editing, Resources, Project administration, Investigation, Funding acquisition, Conceptualization. **Kayvan Pazouki:** Writing – review & editing, Supervision, Project administration, Methodology, Investigation, Funding acquisition, Conceptualization. **Rose Norman:** Writing – original draft, Supervision, Investigation, Formal analysis, Conceptualization.

Declaration of competing interest

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

Appendix 1. . The dynamic weather data at different timestamps

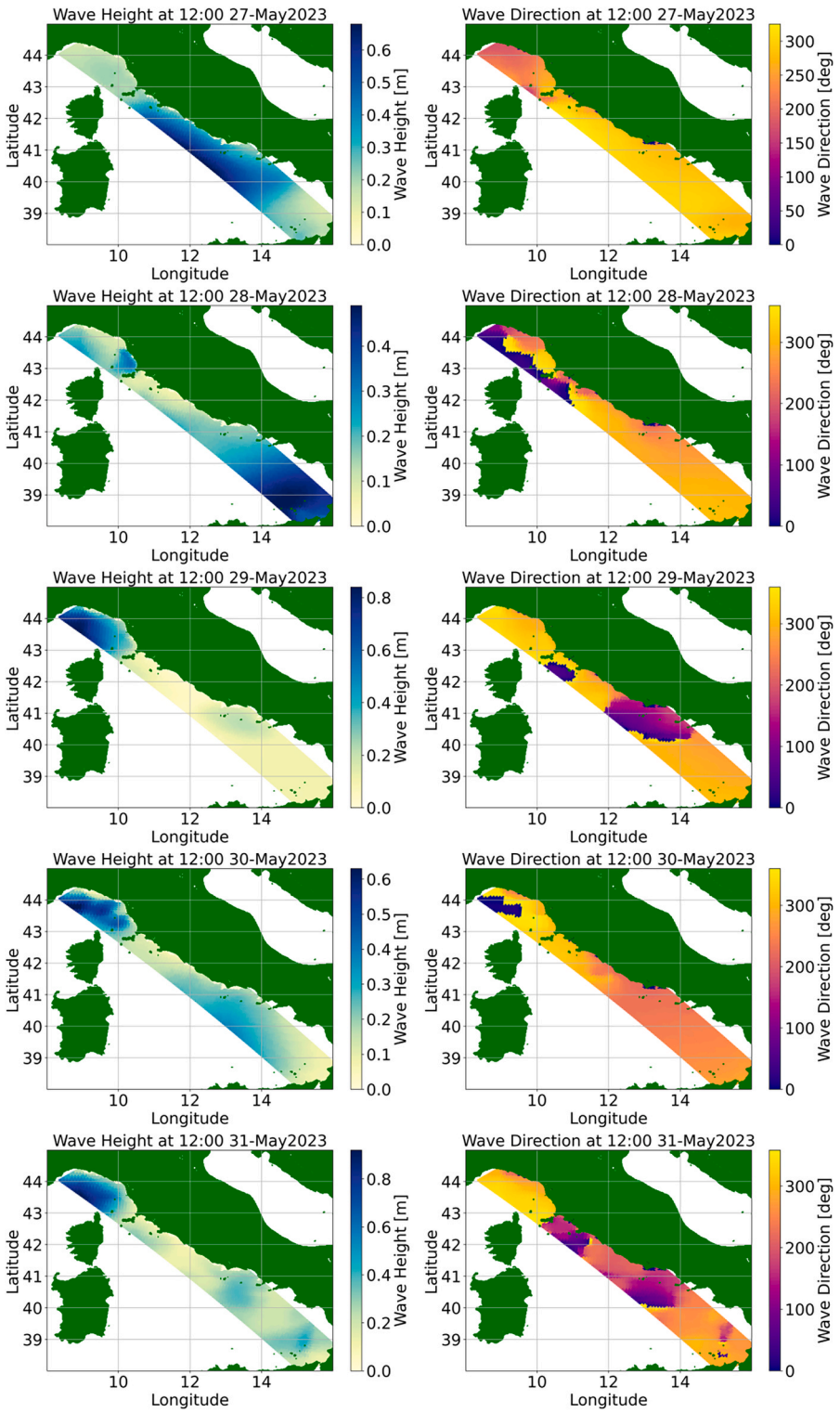


Fig. A1. The time-based wave data

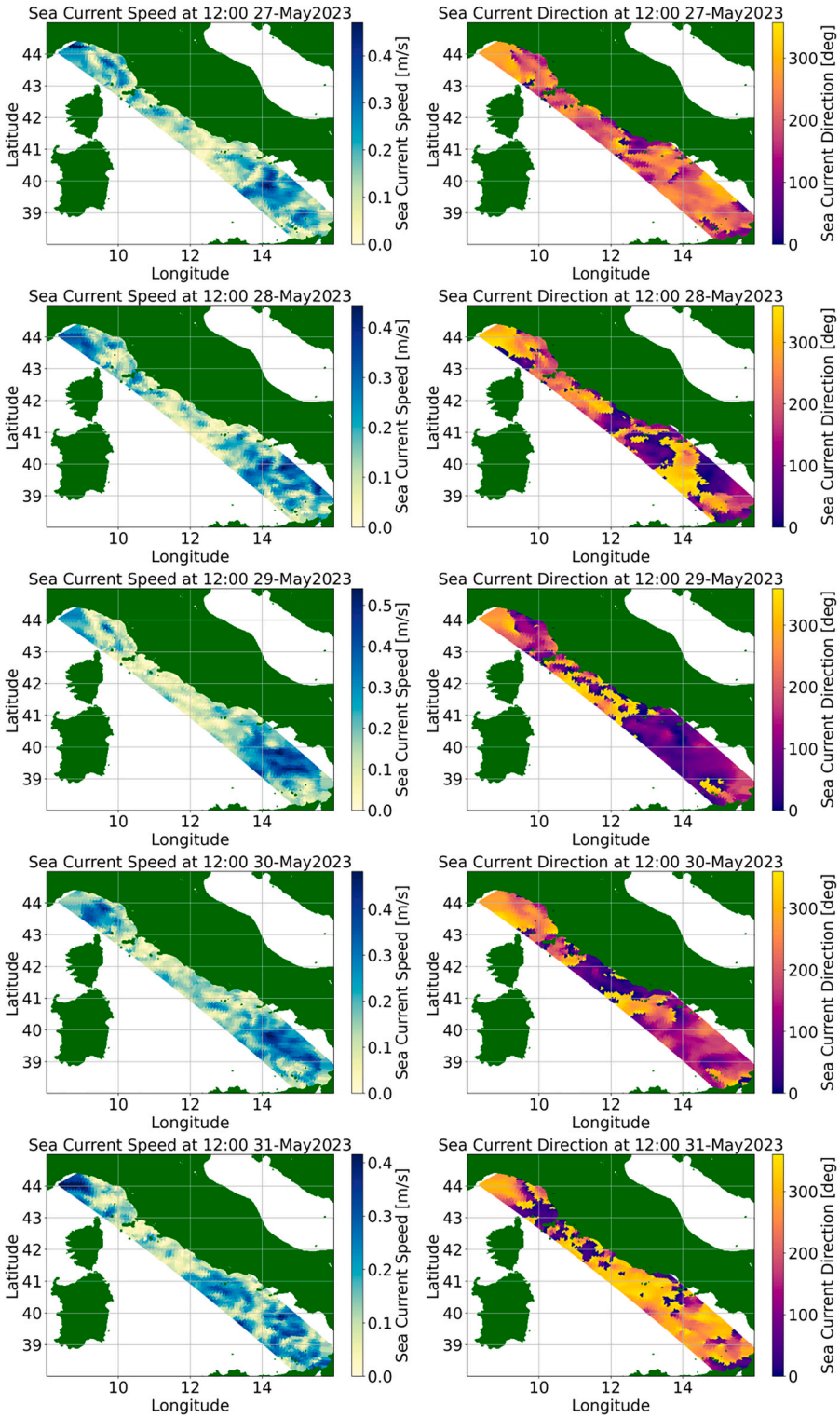


Fig. A2. The time-based current data

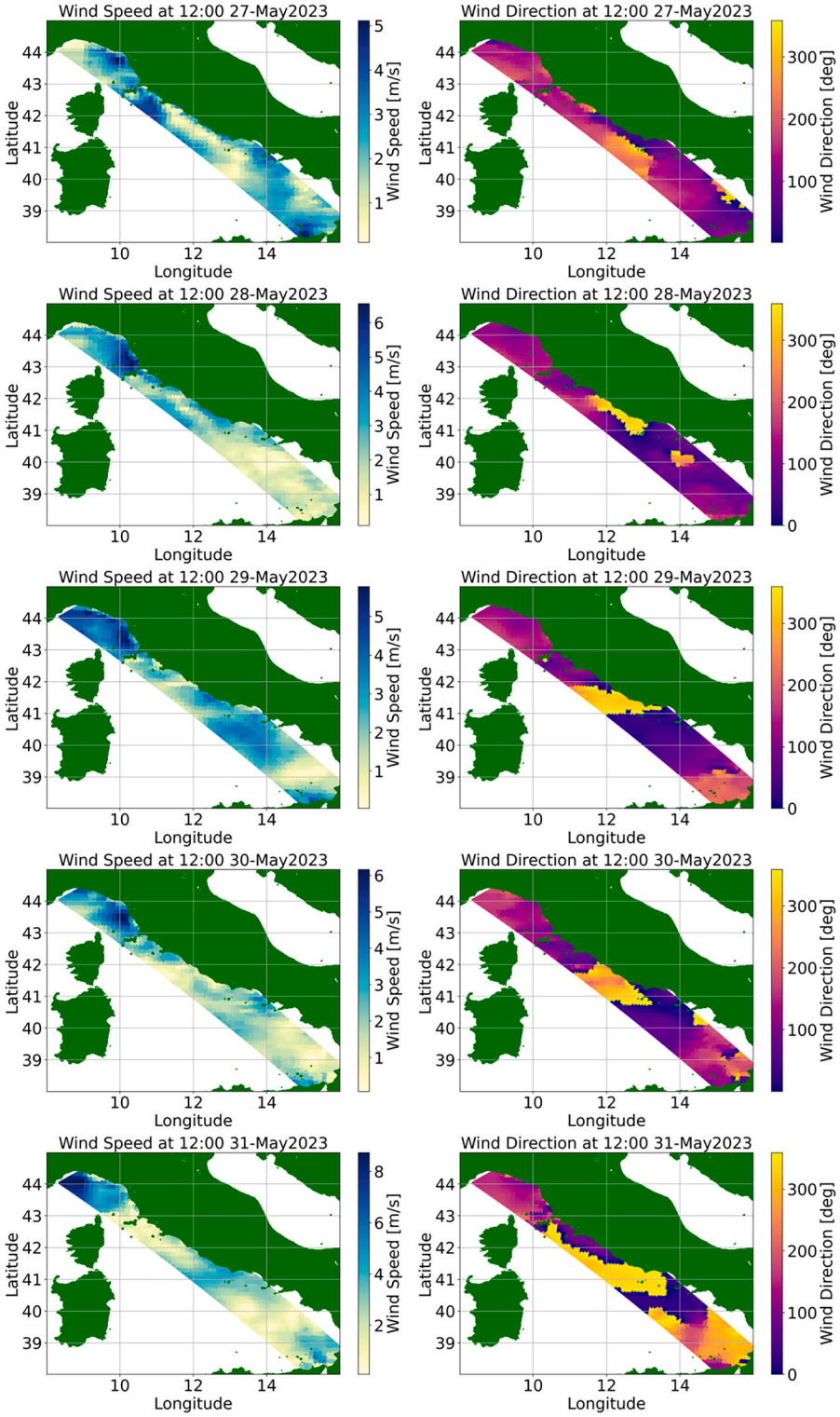


Fig. A3. The time-based wind data

Appendix 2. . Specification of enhanced DQN, standard DQN and PSO models

.

Table A1
Enhanced DQN and Standard DQN Models.

Parameter	Enhanced DQN	Standard DQN
State Size	13	13
Action Size	3–5 directions × 41 speeds	3–5 directions × 41 speeds
Hidden Layers	2	2
Hidden Units	128, 128	128, 128
Activation Function	ReLU	ReLU
Optimizer	Adam	Adam
Learning Rate	0.001	0.001
Discount Factor (γ)	0.99	0.99
Epsilon Start (ϵ)	1.0	1.0
Epsilon End (ϵ_{min})	0.01	0.01
Epsilon Decay Rate	0.999	0.998
Replay Memory Type	Prioritized Experience Replay	Standard Experience Replay
Replay Memory Size	10,000	N/A
Prioritization Alpha	0.6	N/A
Importance Sampling Beta Start	0.4	N/A
Beta Annealing Frames	200,000	N/A
Batch Size	64	64
N-Step Returns	5	N/A
Loss Function	Mean Squared Error (MSE)	Mean Squared Error (MSE)
Number of Training Episodes	250 episodes	250 episodes

Table A2
PSO Models.

Parameter	PSO
Number of Steps	177
Number of Particles	20
PSO Inertia Weight (ω_{PSO})	0.9
PSO Cognitive Coefficient ($c_{1,PSO}$)	0.5
PSO Social Coefficient ($c_{2,PSO}$)	0.3
Number of Training Episodes	250

Table A3
Reward Function.

Parameter	Value
Penalty weight for an invalid move ($\omega_{invalid\_move}$)	1000
Penalty weight for deviation from target step time (ω_{time})	125
Target (maximum acceptable) step time in hours (t_{max})	49.34/177

Data availability

The ship data for FOC model development are confidential; metocean and geographic data are available from CMEMS and NOAA.

References

Beşikçi, E.B., Arslan, O., Turan, O., Ölçer, A.I., 2016. An artificial neural network based decision support system for energy efficient ship operations. Comput. Oper. Res. 66, 393–401. https://doi.org/10.1016/j.cor.2015.04.004.

Calvert, S., 1990. *Optimal weather routeing procedures for vessels on trans-oceanic voyages*. University of Plymouth. Doctoral dissertation.

Carchen, A., Turkmen, S., Piaggio, B., Shi, W., Sasaki, N., Atlar, M., 2021. Investigation of the manoeuvrability characteristics of a Gate Rudder system using numerical, experimental, and full-scale techniques. Appl. Ocean Res. 106, 102419. https://doi.org/10.1016/j.apor.2020.102419.

Castresana, J., Gabiña, G., Quincoces, I., Uriondo, Z., 2023. Healthy marine diesel engine threshold characterisation with probability density functions and ANNs. Reliab. Eng. Syst. Saf. 238, 109466. https://doi.org/10.1016/j.ress.2023.109466.

Chen, T., Guestrin, C., 2016. Xgboost: a scalable tree boosting system. In: *Proceedings of the 22nd Acm Sigkdd International Conference on Knowledge Discovery and Data Mining*, pp. 785–794. https://doi.org/10.1145/2939672.2939785.

Chen, C., Chen, X.Q., Ma, F., Zeng, X.J., Wang, J., 2019. A knowledge-free path planning approach for smart ships based on reinforcement learning. Ocean Eng. 189, 106299. https://doi.org/10.1016/j.oceaneng.2019.106299.

Chen, X., Lv, S., Shang, W.L., Wu, H., Xian, J., Song, C., 2024. Ship energy consumption analysis and carbon emission exploitation via spatial-temporal maritime data. Appl. Energy 360, 122886. https://doi.org/10.1016/j.apenergy.2024.122886.
Chen, Z.S., Lam, J.S.L., Xiao, Z., 2023. Prediction of harbour vessel fuel consumption based on machine learning approach. Ocean Eng. 278, 114483. https://doi.org/10.1016/j.oceaneng.2023.114483.
Chicco, D., Warrens, M.J., Jurman, G., 2021. The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation. PeerJ Comput. Sci. 7, e623.
Chung, N., Balaji, G.N., Rudzki, K., Hoang, A.T., 2025. Internet of things-driven approach integrated with explainable machine learning models for ship fuel consumption prediction. Alex. Eng. J. 118, 664–680. https://doi.org/10.1016/j.aej.2025.01.067.
Coraddu, A., Oneto, L., Baldi, F., Anguita, D., 2017. Vessels fuel consumption forecast and trim optimisation: a data analytics perspective. Ocean Eng. 130, 351–370. https://doi.org/10.1016/j.oceaneng.2016.11.058.
Du, W., Li, Y., Shi, J., Sun, B., Wang, C., Zhu, B., 2023. Applying an improved particle swarm optimization algorithm to ship energy saving. Energy 263, 126080. https://doi.org/10.1016/j.energy.2022.126080.
Du, W., Li, Y., Zhang, G., Wang, C., Zhu, B., Qiao, J., 2022a. Ship weather routing optimization based on improved fractional order particle swarm optimization. Ocean Eng. 248, 110680. https://doi.org/10.1016/j.oceaneng.2022.110680.
Du, Y., Chen, Y., Li, X., Schönborn, A., Sun, Z., 2022b. Data fusion and machine learning for ship fuel efficiency modeling: Part II–Voyage report data, AIS data and meteorological data. Commun. Transp. Res. 2, 100073. https://doi.org/10.1016/j.commtr.2022.100073.
Endresen, Ø., Sørgård, E., Sundet, J.K., Dalsøren, S.B., Isaksen, I.S., Berglen, T.F., Gravir, G., 2003. Emission from international sea transportation and environmental impact. J. Geophys. Res. Atmos. 108 (D17). https://doi.org/10.1029/2002JD002898.
Engelbrecht, A.P., 2007. Computational intelligence: an introduction. John Wiley & Sons.
Eyring, V., Isaksen, I.S., Berntsen, T., Collins, W.J., Corbett, J.J., Endresen, O., Stevenson, D.S., 2010. Transport impacts on atmosphere and climate: shipping. Atmos. Environ. 44 (37), 4735–4771. https://doi.org/10.1016/j.atmosenv.2009.04.059.
Faisal, S., Tutz, G., 2021. Multiple imputation using nearest neighbor methods. Inf. Sci. 570, 500–516. https://doi.org/10.1016/j.ins.2021.04.009.
Fan, A., Yang, J., Yang, L., Wu, D., Vladimir, N., 2022. A review of ship fuel consumption models. Ocean Eng. 264, 112405. https://doi.org/10.1016/j.oceaneng.2022.112405.
Gibson, M., Murphy, A.J., Pazouki, K., 2019. Evaluation of environmental performance indices for ships. Transp. Res. Part D: Transp. Environ. 73, 152–161. https://doi.org/10.1016/j.trd.2019.07.002.
Gkerekos, C., Lazakis, I., Theotokatos, G., 2019. Machine learning models for predicting ship main engine fuel Oil Consumption: a comparative study. Ocean Eng. 188, 106282. https://doi.org/10.1016/j.oceaneng.2019.106282.
Gkerekos, C., Lazakis, I., 2020. A novel, data-driven heuristic framework for vessel weather routing. Ocean Eng. 197, 106887. https://doi.org/10.1016/j.oceaneng.2019.106887.
Global Monitoring and Forecasting Center., 2019a. Global ocean 1/12 physics analysis and forecast updated daily product. EU Copernicus Marine Service Information [Data set]. Available at: https://doi.org/10.48670/moi-00016 (Accessed: 25 Aug 2024).
Global Monitoring and Forecasting Center., 2019b. Global Ocean Waves Analysis and Forecast. E.U. Copernicus Marine Service Information . Available at: https://doi.org/10.48670/moi-00017 (Accessed: 25 Aug 2024).
Monitoring, G., Center, F., 2022. Hourly global ocean sea surface wind and stress from scatterometer and model product WIND_GLO_PHY_L4_NRT_012_004. Available at: Https://. https://doi.org/10.48670/moi-00305. Accessed: 25 Aug 2024.
Hagiwara, H., 1989. Weather routing of (sail-assisted) motor vessels. PhD Thesis. Technical University of Delft. Retrieved from. https://ci.nii.ac.jp/naid/20000749977/.
Henderson, P., Islam, R., Bachman, P., Pineau, J., Precup, D., & Meger, D., 2018, April. Deep reinforcement learning that matters. In *Proceedings of the AAAI conference on artificial intelligence* (Vol. 32, No. 1). https://doi.org/10.1609/aaai.v32i1.11694.
Hessel, M., Modayil, J., Van Hasselt, H., Schaul, T., Ostrovski, G., Dabney, W., ... & Silver, D., 2018, April. Rainbow: Combining improvements in deep reinforcement learning. In *Proceedings of the AAAI conference on artificial intelligence* (Vol. 32, No. 1). https://doi.org/10.1609/aaai.v32i1.11796.
Hoang, A.T., Foley, A.M., Nižetić, S., Huang, Z., Ong, H.C., Ölçer, A.I., Nguyen, X.P., 2022. Energy-related approach for reduction of CO2 emissions: a critical strategy on the port-to-ship pathway. J. Clean. Prod. 355, 131772. https://doi.org/10.1016/j.jclepro.2022.131772.
Hoang, A.T., Pandey, A., De Osés, F.J.M., Chen, W.H., Said, Z., Ng, K.H., Nguyen, X.P., 2023. Technological solutions for boosting hydrogen role in decarbonization strategies and net-zero goals of world shipping: challenges and perspectives. Renew. Sustain. Energy Rev. 188, 113790. https://doi.org/10.1016/j.rser.2023.113790.
Hu, Z., Fan, A., Mao, W., Shu, Y., Wang, Y., Xia, M., Li, B., 2025. Ship energy consumption prediction: Multi-model fusion methods and multi-dimensional performance evaluation. Ocean Eng. 322, 120538. https://doi.org/10.1016/j.oceaneng.2025.120538.
Ioffe, S., Szegedy, C., 2015. Batch normalization: accelerating deep network training by reducing internal covariate shift. In: *International Conference on Machine Learning*. pmlr, pp. 448–456.
Kaklis, D., Kontopoulos, I., Varlamis, I., Emiris, I.Z., Varelas, T., 2024. Trajectory mining and routing: a cross-sectoral approach. Journal of Marine Science and Engineering 12 (1), 157. https://doi.org/10.3390/jmse12010157.
Karagiannidis, P., Themelis, N., Zaraphonitis, G., Spandonidis, C., Giordamlis, C., 2019. November. Ship fuel consumption prediction using artificial neural networks. In: *Proceedings of the Annual Meeting of Marine Technology Conference Proceedings*, pp. 46–51.
Kennedy, J., Eberhart, R., 1995. Particle swarm optimization. In: *Proceedings of ICNN'95-International Conference on Neural Networks*. ieee, pp. 1942–1948. https://doi.org/10.1109/ICNN.1995.488968.
Lee, S.M., Roh, M.I., Kim, K.S., Jung, H., Park, J.J., 2018. Method for a simultaneous determination of the path and the speed for ship route planning problems. Ocean Eng. 157, 301–312. https://doi.org/10.1016/j.oceaneng.2018.03.068.
Li, X., Du, Y., Chen, Y., Nguyen, S., Zhang, W., Schönborn, A., Sun, Z., 2022. Data fusion and machine learning for ship fuel efficiency modeling: Part I–Voyage report data and meteorological data. Commun. Transp. Res. 2, 100074. https://doi.org/10.1016/j.commtr.2022.100074.
Li, Z., Wang, K., Hua, Y., Liu, X., Ma, R., Wang, Z., Huang, L., 2024. GA-LSTM and NSGA-III based collaborative optimization of ship energy efficiency for low-carbon shipping. Ocean Eng. 312, 119190. https://doi.org/10.1016/j.oceaneng.2024.119190.
Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., ... & Wierstra, D., 2015. Continuous control with deep reinforcement learning. *arXiv preprint arXiv: 1509.02971*. https://doi.org/10.48550/arXiv.1509.02971.
Lin, Y.H., Fang, M.C., Yeung, R.W., 2013. The optimisation of ship weather-routing algorithm based on the composite influence of multi-dynamic elements. Appl. Ocean Res. 43, 184–194. https://doi.org/10.1016/j.apor.2013.07.010.
Lindstad, E., Lagemann, B., Rialland, A., Gamlem, G.M., Valland, A., 2021. Reduction of maritime GHG emissions and the potential role of E-fuels. Transp. Res. Part D: Transp. Environ. 101, 103075. https://doi.org/10.1016/j.trd.2021.103075.
Lu, R., Turan, O., Boulougouris, E., Banks, C., Incecik, A., 2015. A semi-empirical ship operational performance prediction model for voyage optimisation towards energy efficient shipping. Ocean Eng. 110, 18–28. https://doi.org/10.1016/j.oceaneng.2015.07.042.
Mannarini, G., Salinas, M.L., Carelli, L., Petacco, N., Orović, J., 2024. VISIR-2: ship weather routing in Python. Geosci. Model Dev. 17 (10), 4355–4382. https://doi.org/10.5194/gmd-17-4355-2024.
Marie, S., Courteille, E., 2009. 1 Multi-objective optimisation of motor vessel route. In: Marine Navigation and Safety of Sea Transportation. CRC Press, pp. 437–444.
MEPC., 2008. Amendments to the technical code on control of emission of nitrogen oxides from marine diesel engines. International Maritime Organization. Retrieved from https://wwwcdn.imo.org/localresources/en/OurWork/Environment/Documents/177(58).pdf.
MEPC, R., 2023. 2023 IMO strategy on reduction of GHG emissions from ships.
Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Hassabis, D., 2015. Human-level control through deep reinforcement learning. Nature 518 (7540), 529–533. https://doi.org/10.1038/nature14236.

Moradi, M.H., Brutsche, M., Wenig, M., Wagner, U., Koch, T., 2022. Marine route optimisation using reinforcement learning approach to reduce fuel consumption and consequently minimize CO2 emissions. Ocean Eng. 259, 111882. https://doi.org/10.1016/j.oceaneng.2022.111882.
NOAA National Geophysical Data Center., 2009. ETOPO1 1 Arc-Minute Global Relief Model. NOAA National Centers for Environmental Information. . Available at: https://www.ncei.noaa.gov/access/metadata/landing-page/bin/iso?id=gov.noaa.ngdc.mgg.dem:316 (Accessed: 25 Aug 2024).
Opdam, W., & Bijlard, M., 2024. Working Principle of a Gate Rudder.
Papandreou, C., Ziakopoulos, A., 2022. Predicting VLCC fuel consumption with machine learning using operationally available sensor data. Ocean Eng. 243, 110321. https://doi.org/10.1016/j.oceaneng.2021.110321.
Sasaki, N., Atlar, M., 2018. November. Investigation into the propulsive efficiency characteristics of a ship with the Gate Rudder propulsion. *A. Yücel Odabaşı Colloquium Series3rd International Meeting on Progress in Propeller Cavitation and Its Consequences: Experimental and Computational Methods for Predictions*.
Shao, W., Zhou, P., 2011. Development of a dynamic programming method for low fuel consumption and low carbon emission from shipping. *International Conference on Technologies, Operation and Logistics and Modelling for Low Carbon Shipping*.
Simonsen, M., Walnum, H.J., Gössling, S., 2018. Model for estimation of fuel consumption of cruise ships. Energies 11 (5), 1059. https://doi.org/10.3390/en11051059.
Tillig, F., Ringsberg, J.W., 2019. A 4 DOF simulation model developed for fuel consumption prediction of ships at sea. Ships Offshore Struct. 14 (sup1), 112–120. https://doi.org/10.1080/17445302.2018.1559912.
Trodden, D.G., Murphy, A.J., Pazouki, K., Sargeant, J., 2015. Fuel usage data analysis for efficient shipping operations. Ocean Eng. 110, 75–84. https://doi.org/10.1016/j.oceaneng.2015.09.028.
Turkmen, S., Wang, L., Raftopoulos, S., Li, C., Norman, R., 2023. Analysis of the hydrodynamic performance of a gate rudder system. In: *IOP Conference Series: Materials Science and Engineering*. IOP Publishing. https://doi.org/10.1088/1757-899X/1288/1/012059. No. 1.
Uyanık, T., Karatuğ, Ç., Arslanoğlu, Y., 2020. Machine learning approach to ship fuel consumption: a case of container vessel. Transp. Res. Part D: Transp. Environ. 84, 102389. https://doi.org/10.1016/j.trd.2020.102389.
Van Hasselt, H., Guez, A., Silver, D., 2016. Deep reinforcement learning with double q-learning. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. https://doi.org/10.1609/aaai.v30i1.10295. No. 1.
Vettor, R., Soares, C.G., 2016. Development of a ship weather routing system. Ocean Eng. 123, 1–14. https://doi.org/10.1016/j.oceaneng.2016.06.035.
Walsh, C., Bows, A., 2012. Size matters: exploring the importance of vessel characteristics to inform estimates of shipping emissions. Appl. Energy 98, 128–137. https://doi.org/10.1016/j.apenergy.2012.03.015.
Wang, H., Lang, X., Mao, W., 2021. Voyage optimisation combining genetic algorithm and dynamic programming for fuel/emissions reduction. Transp. Res. Part D: Transp. Environ. 90, 102670. https://doi.org/10.1016/j.trd.2020.102670.
Wang, H., Mao, W., Eriksson, L., 2019. A Three-Dimensional Dijkstra's algorithm for multi-objective ship voyage optimisation. Ocean Eng. 186, 106131. https://doi.org/10.1016/j.oceaneng.2019.106131.
Wang, Z., Schaul, T., Hessel, M., Hasselt, H., Lanctot, M., Freitas, N., 2016. Dueling network architectures for deep reinforcement learning. In: *International Conference on Machine Learning*. PMLR, pp. 1995–2003.
Wieczorek, J., Guerin, C., McMahon, T., 2022. K-fold cross-validation for complex sample surveys. Stat 11 (1), e454.
Xie, X., Sun, B., Li, X., Zhao, Y., Chen, Y., 2023. Joint optimization of ship speed and trim based on machine learning method under consideration of load. Ocean Eng. 287, 115917. https://doi.org/10.1016/j.oceaneng.2023.115917.
Yao, J., Luo, X., Li, F., Li, J., Dou, J., Luo, H., 2024. Research on hybrid strategy Particle Swarm Optimization algorithm and its applications. Sci. Rep. 14 (1), 24928. https://doi.org/10.1038/s41598-024-76010-y.
Zaccone, R., Ottaviani, E., Figari, M., Altosole, M., 2018. Ship voyage optimisation for safe and energy-efficient navigation: a dynamic programming approach. Ocean Eng. 153, 215–224. https://doi.org/10.1016/j.oceaneng.2018.01.100.
Zhang, M., Tsoulakos, N., Kujala, P., Hirdaris, S., 2024. A deep learning method for the prediction of ship fuel consumption in real operational conditions. Eng. Appl. Artif. Intel. 130, 107425. https://doi.org/10.1016/j.engappai.2023.107425.
Zhou, T., Hu, Q., Hu, Z., Zhen, R., 2022. An adaptive hyper parameter tuning model for ship fuel consumption prediction under complex maritime environments. J. Ocean. Eng. Sci. 7 (3), 255–263. https://doi.org/10.1016/j.joes.2021.08.007.
Zhou, Y., Pazouki, K., Murphy, A.J., Uriondo, Z., Granado, I., Quincoces, I., Fernandes-Salvador, J.A., 2023. Predicting ship fuel consumption using a combination of metocean and on-board data. Ocean Eng. 285, 115509. https://doi.org/10.1016/j.oceaneng.2023.115509.
Zhou, Y., Pazouki, K., Norman, R., 2025. A grey-box deep learning modelling strategy for fuel oil consumption prediction: a case study of tuna purse seiner. Ocean Eng. 324, 120733. https://doi.org/10.1016/j.oceaneng.2025.120733.
Zhu, T., Naseri, M., Dhar, S., Ashrafi, B., 2024. Reinforcement learning for fishing route planning and optimisation. *International Conference on Offshore Mechanics and Arctic Engineering*. American Society of Mechanical Engineers.