applied sciences

MDPI

Article

An Efficient Maritime Route Planning Method Based on an Improved A* with an Adaptive Heuristic Function and Parallel Computing Structure

Hanlin Li [1], Longxia Qian [1,*], Mei Hong [2,*], Xianyue Wang [1] and Zilong Guo [2]

[1] School of Science, Nanjing University of Posts and Telecommunications, Nanjing 210023, China; lihanlin.zzz@foxmail.com (H.L.); wxy278499177@163.com (X.W.)
[2] College of Meteorology and Oceanography, National University of Defense Technology, Changsha 410073, China; gzlong_@nudt.edu.cn
* Correspondence: qianlongxia@njupt.edu.cn (L.Q.); flowerrainhm@126.com (M.H.)

Abstract: Maritime route planning under minimal-risk conditions plays an important part in the development and utilization of marine resources. High-resolution weather forecasting data places higher demands on the algorithms' ability to optimize and compute, and existing algorithms are significantly deficient in these aspects. Therefore, we propose a parallel computing-based planning method, segment parallel A* (SPA*), which splits the path into small segments and runs A* separately on CPU cores through a control algorithm. In segment planning, we propose an adaptive heuristic function on A*. It automatically balances the order of magnitude difference between the risk assessment value and the estimated distance, thus significantly reducing the A* expansion useless grid to improve the performance and running speed of the algorithm. Finally, the complete route is obtained by splicing the above segments. In the static planning experiments, the time of SPA* is reduced by about 5~12,425 times compared with 6 traditional and swarm intelligence-based algorithms, i.e., Dijkstra, A*, bidirectional A* (BA*), ant colony optimization (ACO), Harris hawks optimization (HHO), and sparrow search algorithm (SSA). And the abilities to control the risk caused by wind and waves and the comprehensive risk are improved by 7.68%~25.14% and 8.44%~14.38%, respectively; in the dynamic planning experiments, the above results are 4.8~1262.9 times, 3.87%~9.47% and 7.21%~10.36%, respectively. By setting the recommended range of the number of segments for each case, SPA* shows stable performance in terms of the calculation and risk control. SPA* demonstrates a unique structure for using parallel computing in route planning, which is representative and general in both reducing time and improving efficiency.

Keywords: parallel computing; A-star algorithm; weather ship routing; minimum risk route (MRR)

check for updates

Citation: Li, H.; Qian, L.; Hong, M.; Wang, X.; Guo, Z. An Efficient Maritime Route Planning Method Based on an Improved A* with an Adaptive Heuristic Function and Parallel Computing Structure. *Appl. Sci.* **2023**, *13*, 10873. https://doi.org/10.3390/app131910873

Academic Editor: José A. Orosa

Received: 1 September 2023
Revised: 24 September 2023
Accepted: 28 September 2023
Published: 30 September 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

1.1. Background

With the increasing concern for the development of marine resources and maritime rights and interests, the expectation and requirement for intelligent control and autonomous navigation of ships are getting higher and higher, and maritime route planning is directly related to the level of automation and intelligence of ship navigation to a large extent [1–7]. The purpose of the research is to determine how to make the ship better accomplish the tasks specified by the instructions from the upper level and automatically plan the optimal route from the starting point to the specified target position according to certain guidelines. From the viewpoint of the algorithm execution process, maritime route planning is mainly to realize the ship's route search in the obstacle environment, which can be reduced to the problem of optimization search [2,3,8].

However, today there have been important technological breakthroughs in the field of marine meteorological acquisition and monitoring equipment, and civilian, commercial,

and military vessels out on sailing are beginning to have access to high-resolution weather forecast data [8–12]. As a result, all parties are placing higher demands on the planning of routes. They are no longer satisfied with the shortest path provided by the initial route planning system but want to control the various risks on the route as well as the running time of the algorithm [13].

With the maturity and promotion of corresponding standards and norms, more and more papers show that the rules formulated by the International Maritime Organization (IMO) can effectively reduce the risk of shipwreck and improve economic benefits for maritime navigation [14–19]. The IMO points out that the most important factors affecting the safety and efficiency of navigation are the wind and waves along the way [14–16]. It is very necessary to make full use of forecast data to control risk and cost in long-distance navigation. First of all, the research and implementation of IMO standards will help the ship navigation system integrate and optimize with international standards. Different from traditional wave control, the IMO standard is more authoritative, strict and clear [14,15,18]. The existing research shows that wind and waves are the main factors affecting the stability of ships' navigation in the ocean [13,20]. When meeting the IMO standard, risk control under severe sea conditions is particularly prominent, such as the avoidance and control of dangerous navigation phenomena such as wave riding, swaying, and parametric rolling. Secondly, the problem of increased computational complexity caused by IMO standards, high-resolution data, and long-distance navigation is also a pain point in the field of path planning. Addressing this challenge will also expand new optimization directions and ideas for path-planning algorithms. In summary, the research on IMO and high-resolution path planning is of great significance. Therefore, how to carry out long-distance route planning with wind and wave constraints on high-resolution grid data under the IMO standard is the focus of the continuous attention of shipping parties.

1.2. Literature Review

Internationally, the mainstream path planning methods include: Dijkstra [21], A* [13,15,16,20] and its variant bidirectional A* (BA*) [22,23] based on node search; path planning based on ant colony optimization (ACO) [24,25], Harris hawks optimization (HHO) [26,27] and sparrow search algorithm (SSA) [28–30]; and path planning based on Q-Learning. At present, many scholars have applied IMO to maritime route planning. Some scholars have achieved some results by using the Martin algorithm [14], which provides a reference for applying IMO rules to route planning. Its principle is to search and optimize through preset rows of nodes pointing to the end point. Other scholars use A * to plan routes under the standard of IMO [13,15,16]. Some scholars also introduce a swarm intelligence algorithm to solve the problem of route planning when the amount of data is small [21,24,26]. However, the above algorithms still have the following defects. A class represented by Martin, due to the preset node arrangement, is very sparse; although the algorithm's running time is relatively short, the planning route is fuzzy. It is difficult to achieve theoretical results by projecting such routes into actual navigation scenarios; the calculation time of Dijkstra and A * is unacceptable to the industry because of their low search efficiency due to the rapid increase in data volume. However, based on the comparison of the results of multiple papers, the path they planned has high accuracy and is better applied in actual scenarios. The swarm intelligence algorithm is to complete the initialization of the population according to certain rules and continuously update the population through iteration to obtain the optimal solution to the problem. Maritime route planning is a non-deterministic polynomial (NP) hard problem just like route planning. The swarm intelligence algorithms have obvious advantages in solving most of the NP hard problems, but there are still some problems in solving the grid shortest path problem. When the number of grids is large, its performance is relatively unstable. We found that the main reason is that the swarm intelligence algorithm plans the key points of the path first and then connects all the key points into the planned path correctly using a connection algorithm. The performance of the connection algorithm will be severely affected when the

number of grids increases. There are still many problems in the application of the swarm intelligence path planning algorithm, such as early convergence, falling into a local optimal solution, too many parameters to be adjusted, and a large amount of calculation for the grid data.

Similarly, the current path planning based on Q-Learning [31,32] is still subject to many obstacles in the application, such as the need for a huge amount of simulation and feedback to generate the path. When considering the synergy of multi-agents, Q-Learning can achieve a comprehensive and better effect. But in the path planning of only one individual, scholars prefer to use traditional path planning. In summary, the current engineering aspects believe that route planning is more effective using some variant algorithms based on A*.

The essence of solving long-distance route planning under IMO and high-resolution standards is to improve the performance and efficiency of the algorithm. In the process of continuous planning of the algorithm, a large amount of grid data is continuously sent to the planning terminal according to the time period. At the same time, the algorithm also needs to retrieve whether the grid meets some quantitative suggestions in IMO, which makes the traditional algorithm and related improved algorithms insufficient to support its normal operation, and the time consumption is multiplied. It is noted that parallel technology is an effective means to improve the performance and efficiency of the algorithm. At present, parallel allocation and computer mechanisms under various structures and mechanisms are gradually introduced into machine learning and swarm intelligence optimization algorithms [33,34]. In the field of path planning involving machine learning and swarm intelligence optimization, adding parallel computing structures is also gradually forming a trend.

However, at present, most of them are optimized for generating paths with the shortest path property, and the field of maritime route planning focuses on obtaining a minimum-risk path. This difference requires complex migration for parallel traditional algorithms to be applied normally. For example, bidirectional A* is an improvement based on the idea of parallel computing. It searches from the starting point and the end point at the same time. When the search nodes in both directions overlap, the algorithm stops and outputs the shortest path. However, for the traditional shortest path, the two searches that start and end simultaneously do not affect the performance of the shortest path but improve the efficiency of planning. For the minimum-risk route, the wind direction and wave direction along the route need to be checked. At this time, the search from the end point and the search from the start point are different based on the relative angles of the wind and wave fields. Even if the corresponding coordinate system reversal and equivalent operation are performed, due to the node expansion mechanism affected by the heuristic function in A*, the performance of the generated risk-minimization route has a large loss compared with the performance of A*. Therefore, directly using the above parallel algorithm in maritime route planning will greatly reduce the performance of the obtained path.

1.3. Motivation, Technical Characterization and Contribution

In summary, existing maritime path planning algorithms suffer from low accuracy and low operational efficiency, which makes the algorithms unable to support more constraints; parallel computing has been successfully used to improve algorithmic performance in the shortest path problem, but it is seldom seen in maritime minimum-risk path planning. Aiming at the above problems, this paper proposes a path planning method based on parallel computing, namely segment parallel A* (SPA*). Its technical process and characteristics are shown in Figure 1. SPA* needs to carry out a relevant risk assessment of the sea area where the route is located based on prior information, such as terrain risk, wind wave risk, and comprehensive risk. Firstly, a standard path with minimum terrain risk is generated based on A*. Secondly, the standard path is divided equally. The segments obtained by dividing the standard path are assigned to the core of the CPU through a control algorithm to run A*, respectively. In this step, we propose an adaptive heuristic function for A*. It is used

to reduce A* expansion to useless grids, thereby improving algorithm performance and running speed. Finally, the planned track of each segment is spliced into a minimum-risk track from the starting point to the end point.

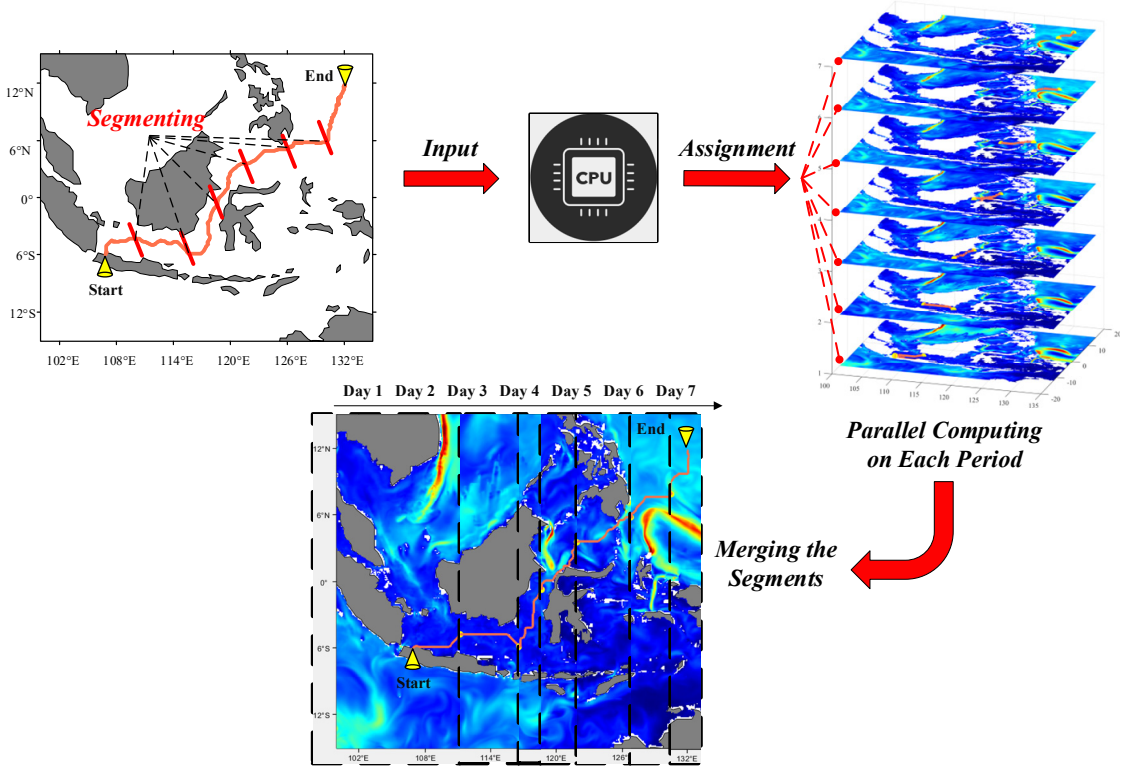

Figure 1. The overview chart on the technical characteristics of SPA*.

The main contributions of this paper are: (1) For long-distance planning tasks, multi-constraint conditions, and high-resolution data requirements, SPA* greatly reduces computational time while maintaining performance through parallel allocation. Compared with some current parallel path planning algorithms, it is more compatible. more compatible. Whether it is the problem of the minimum-risk route or the problem of the shortest path, SPA* is easy to adjust and use directly. (2) The algorithm considers more complex meteorological conditions. We consider the stalling of ships due to wind and wave fields and propose vector risk and two constraints on IMO. This makes the sea route planning more in line with the norms of authoritative organizations and closer to reality.

2. Models

2.1. The Gridded Path Constraint Model Based on the International Maritime Organization (IMO)

Environmental map is the premise of the path planning task. In general, the planning of the ship's track requires electronic charts to provide global environmental information. The environment map is usually composed of complex geometric figures, which means most of the search algorithms cannot be directly applied in path planning, and the scale of the map directly affects the efficiency of path planning, so it is necessary to process the environment map. The grid map is the most commonly used form of processing and expression. It represents the environment as a square grid network with a certain resolution. The grids are divided into free state and occupied state according to whether they are occupied by obstacles, and the grids are independent of each other. The grid map is easy to create and easy to implement path planning, so it is widely used in robotics, autonomous driving, and other fields.

The IMO points out that the constituents of environmental risk during the navigation of a ship at sea are mainly wind and waves. Based on this, many scholars have transformed the IMO's quantified limitation rules for wind and waves into constraints in the route planning algorithm in grid maps. Scholars point out that one of the factors of the ship most affected by wind and wave conditions is its operating speed. Winds and waves with different angles and strengths can cause ships to stall at different levels. The stall can lead to various dangerous phenomena on the ship, among which the more popular ones are surf-riding and broaching-to. For a long time, many ship trajectory planning considered only distance cost and scalar risk cost. In this paper, we note the scalar risk assessment array as F_1 and the vector risk assessment array as F_2. F_1 and F_2 are 3D arrays, with the first two dimensions being longitude and latitude, and the third dimension being time.

The average risk value of the grid through which the route passes is taken as f_1 by evaluating the scalar data from the ocean forecast, and f_1 is often normalized to the interval [0, 1]. Traditional ship track planning in terms of risk control focuses on f_1. The assessment methods are the analytical hierarchy process (AHP) [35,36] and fuzzy comprehensive evaluation (FCE) [37]. The main steps of the evaluation are: (1) Establish the FCE factor set U for risk indicators [36,37]. The indicators in this paper are selected as sea surface temperature, sea surface current velocity, and wave height. (2) Use the Gaussian membership function to perform single factor fuzzy evaluation on meteorological data and obtain a fuzzy matrix R. (3) Use AHP combined with expert knowledge to determine A [35,37]. (4) Obtain the fuzzy vector B by $B = A \circ R$, where $\circ$ is the fuzzy synthesis operator and is taken as matrix multiplication in the general case. (5) Use the weighted average principle to determine F_1, which can be expressed as $F_1 = B \cdot (0.2, 0.4, 0.6, 0.8, 1)^T$. Now we assume that the planned path is represented by the planned path array $P = (p_1, p_2, \cdots, p_n)$. $p_i(1)$ is a linear index value representing a specific grid on the map. And $p_i(2)$ is the time period when ships arrive at $p_i(1)$. Linear index values and arrays of row and column coordinates are equivalent. Finally, f_1 is calculated using Equation (1).

$$f_1 = \frac{\sum_{i=1}^{n} F_1(p_i(1), p_i(2))}{n}, \tag{1}$$

where n denotes the number of components in the planned path array P.

However, in actual navigation, wind and waves are particularly influential on surface ships, and both mainly affect the ship's speed through physical effects. For estimating the actual speed of a ship under wind and wave conditions, many scholars and scientific organizations have proposed their own theoretical formulas. In this paper, we adopt one of the empirical formulas obtained from simulations with real measurement data [14–16,38].

$$v = v_0 - (a_1 h - a_2 q h + a_3 v_W \cos\delta) \times (1 - a_4 D v_0), \tag{2}$$

where $a_i, i = 1, 2, 3, 4$ denotes the coefficient determined by the size of the ship. v_0 is assumed to be the constant speed maintained by the surface ship while sailing; q is the angle between the wave direction and the ship's heading; δ is the angle between the wind direction and the ship's heading; D is the displacement of the ship; h is the wave height. Using Equation (2), the actual speed after stall at each extended node can be calculated, and thus the required sailing time for the route can be calculated. For each grid movement, the travel time is obtained by dividing the distance between the grids by the actual speed, which is calculated by Equation (3).

$$G(i) = X_k / v_i, \ (k = 1, 2, 3; \ i = 1, 2, \cdots, n), \tag{3}$$

where $G(i)$ denotes the travel time cost of the ith component on planned path array P; the three values of X_k are the distances between the grids for horizontal, vertical, and diagonal shifts, respectively; n denotes the number of components in the planned path array P. The

travel time is normalized to a dimensionless value of 0 to 1 by the "arctan" function to determine travel time cost $\widetilde{G}$. Then f_2 is calculated using Equation (4).

$$\begin{cases} f_2 = \frac{\sum_{i=1}^{n} \widetilde{G}(i)}{n} \\ \widetilde{G}(i) = \arctan(G(i)) \cdot 2/\pi, i = 1, 2, \cdots, n \end{cases}, \tag{4}$$

where $\widetilde{G}(i)$ represents the travel time cost of the ith component on planned path array P; n denotes the number of components in the planned path array P. According to the International Maritime Organization (IMO) recommendations for avoiding hazards in adverse weather and sea conditions. Ships passing through the grids satisfying constraints one and two below are prone to capsize and cross-sway hazards and should avoid these grids as much as possible. Therefore, hard limits can be imposed on these grids. When the path planning algorithm expands grid j at time k, if grid j does not satisfy any of the above constraints, then it will be temporarily set as impassable. At the beginning of the $k+1$ moment, the restriction is lifted, and the constraint is checked again.

Constraint I: the angle between the wave direction, the ship's heading q, and the actual velocity v in a grid are in the range found in Equation (5).

$$135^\circ < q < 225^\circ \; and \; v > \frac{1.8\sqrt{L_{\text{ship}}}}{\cos(180^\circ - q)}, \tag{5}$$

where L_{ship} is the length of the ship, the ship passing through these grid points will be prone to surf-riding and broaching-to, which increases the possibility of capsizing.

Constraint II: the natural roll period in seconds T_R and the encountered period with waves in seconds T_E satisfy one of the two conditions found in Equation (6).

$$|T_E - T_R| = \varepsilon \cdot T_R \; or \; |2 \cdot T_E - T_R| = \varepsilon \cdot T_R, \tag{6}$$

where ε is the relative tolerance in frequency matching in %.

In summary, the average time cost of the grid passing the route under wind and wave conditions is taken as f_2. Therefore, the model has fully considered scalar risk, vector risk, and IMO constraints on security. The analytic hierarchy process can be used to determine the weight coefficients of f_1 and f_2 in the cost function of A*, and the multi-objective programming problem can be transformed into a single-objective programming problem. In this paper, the weight coefficients of f_1 and f_2 are both 0.5.

2.2. Adaptive Heuristic Function and Segment Parallel A(SPA*)*

As discussed in the introduction, traditional path planning algorithms have found it difficult to cope with long-distance planning tasks, multi-constraint conditions, and high-resolution data requirements. Based on "The gridded path constraint model based on the IMO" established in 2.1, this section introduces one of the two important improvements in this paper. The path planning in this paper defaults to the 8-direction mode. It can be seen in many articles that the directional mode based on the shape of the earth has a 16-direction model and a 48-direction model of ship route planning, which are all reasonable. However, in order to facilitate the demonstration of our model ideas, we chose the classic 8-direction model.

First, we improve the heuristic function in A*. A* is developed based on Dijkstra. Its total cost function takes into account not only the current distance cost but also the predicted distance cost. The total cost function of A* in path planning is described by Equation (7).

$$\begin{gathered} f(cell_i) = g(cell_i) + h(cell_i), \\ i = 1, 2, \cdots, m, \end{gathered} \tag{7}$$

where, $cell_i$ denotes the grid indexed as i on the grid map. $g(cell_i)$ denotes the actual distance cost from the starting grid to the current grid $cell_i$. A* is a path with minimum $\sum_i f(cell_i)$ by establishing the objective function of Equation (7). A* algorithm is very flexible and adaptable and can be designed with specific cost functions for different search tasks. $h(cell_i)$ is also known as the heuristic function. A* can only select a grid with the minimum total cost over all the grids expanded by the heuristic function. It can be said that it can determine the efficiency of the A* search and the performance of the path. Generally, $h(cell_i)$ is estimated using Euclidean distance as the following Equation (8).

$$h(cell_i) \triangleq h(x_i, y_i) = w \cdot \sqrt{(x_i - x_g)^2 + (y_i - y_g)^2}, \tag{8}$$

where, (x_i, y_i) is the row and column coordinates of grid $cell_i$; w is the weight factor that regulates the value of the heuristic function; (x_g, y_g) is the row and column coordinates of the target grid. Now assume that $h*(cell_i)$ is the true cost of grid $cell_i$ to the end grid. But in practice, it is unknowable because it is difficult to know exactly this value until the end point is reached. We can use Table 1 to outline the laws of the heuristic function. If the "Condition" column is satisfied for either grid $cell_i$, then the conclusion of the "Result" column will be obtained.

Table 1. The effect of the setting for the A* heuristic function on the speed and accuracy of the path planning.

Number	Condition	Result
1	$h(cell_i) = 0$	A* degenerates to Dijkstra. The algorithm will expand the grids without difference until it expands to the end grid.
2	$h(cell_i) \leq h*(cell_i)$	A* is guaranteed to find the shortest circuit. And the smaller $h(cell_i)$ is the more grids the algorithm will expand. This causes the algorithm to slow down.
3	$h(cell_i) > h*(cell_i)$	A* is not guaranteed to find the shortest path, but the algorithm will be faster at this point.
4	$h(cell_i) \gg h*(cell_i)$	A* is converted to breath first search (BFS). At this point the algorithm is very fast and will expand the grids as little as possible.

Create a grid map of a real scene as shown in Figure 2. The green color on the diagram represents the island and land, the blue color represents the navigable area, and the yellow color represents the grid points expanded by the algorithm. The red and purple grids represent the starting and ending points, respectively, and the red line represents the A* planning path. Since the risk values f_1 and f_2 are generally normalized to 0~1, the grid-to-grid distance is calculated using great circle sailing (GCS). In this paper, the data resolution is $1/12^\circ$, and the distances between grids in horizontal, vertical, and oblique directions are 4.8321 nautical miles (nm), 5.0026 nm, and 6.9552 nm. It is easy to understand that if $h(cell_i)$ is calculated using Eq. Therefore, this leads to scenario 4 in Table 1. In this paper, $g(cell_i)$ contains both scalar risk and vector risk values, so it is expressed as Equation (9).

$$\begin{aligned} g(cell_i, cell_{i'}) &\triangleq g(x_i, y_i, x_{i'}, y_{i'}), \\ &= \alpha \widetilde{f_1}(x_i, y_i) + (1-\alpha)\widetilde{f_2}(x_i, y_i, x_{i'}, y_{i'}), \alpha \in [0,1], \end{aligned} \tag{9}$$

where $\widetilde{f_1}$ and $\widetilde{f_2}$ denote the row-column coordinate forms of the scalar risk and vector risk defined in 2.1, respectively; $(x_{i'}, y_{i'})$ denotes the rank coordinates of the eight expandable grids around (x_i, y_i); α denotes the weight, with larger values indicating more emphasis on the minimization of f_1; conversely, more emphasis on the minimization of f_2. So traditionally, a static and dynamic approach would be taken to control the coefficients in Equation (9) to make $g(cell_i)$ and $h(cell_i)$ close to one magnitude, thus reaching the possibility of $h(cell_i) = h*(cell_i)$. This way, A* can perform the best.

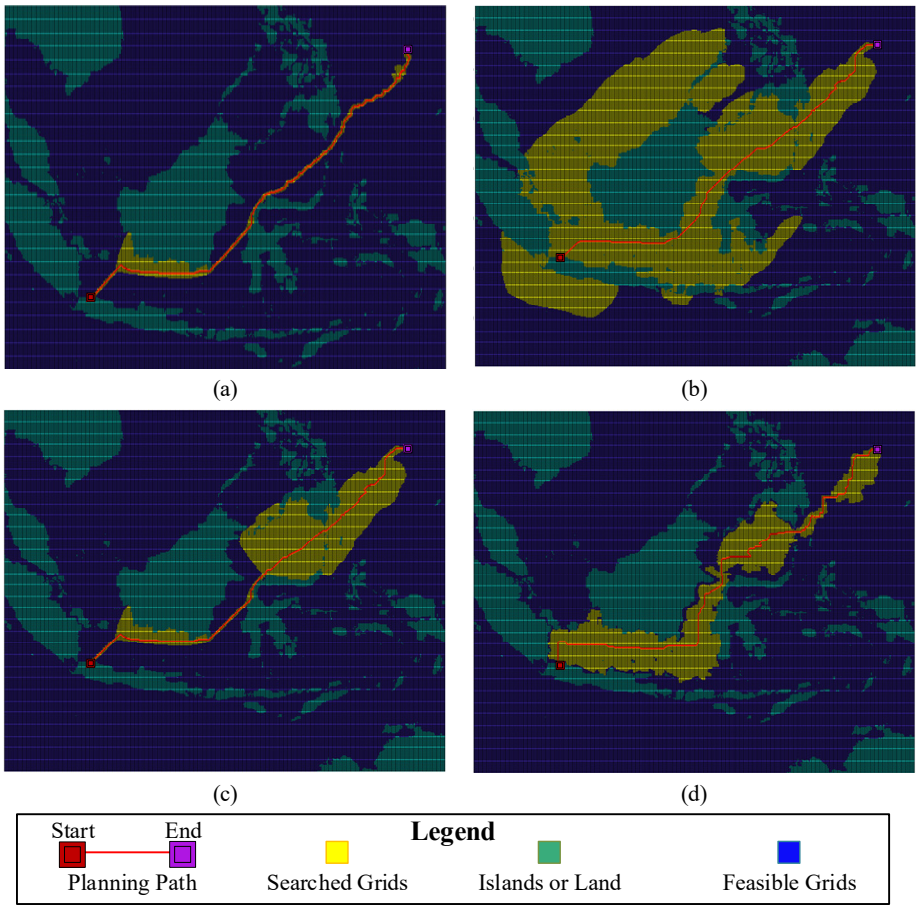

Figure 2. Comparison between the effect of adaptive heuristic function planning and the effect of traditional heuristic function. (**a**) w is 0.3; (**b**) w is 0.15; (**c**) the first half of the route w is 0.3, and the second half of the route w is 0.15. (**d**) w is adjusted by the adaptive heuristic function in this paper.

The static approach is usually to let w be a constant value. Figure 2a,b shows the results obtained by making w 0.3 and 0.15, respectively. It can be seen that Figure 2a corresponds to Table 1 between cases 3 and 4, while Figure 2b corresponds to Table 1 between cases 1 and 2. The number of grids expanded in the former case is obviously not enough to support an optimal route. The latter expands the number of grids to ensure that it obtains an optimal track, and it expands too many useless grids. The dynamic approach usually searches fast in the first half of the algorithm and slows down the search in the second half of the algorithm. Figure 2c shows that the algorithm is designed so that w is 0.3 in the first 50% of the path and 0.15 in the second 50% of the path. It can be observed that the dynamic approach also does not achieve satisfactory results and does not get sufficient search in the first half of the path. In conclusion, it is difficult to achieve satisfactory results with the traditional mechanism of regulating w. It is possible to get a good value of w by continuously trying to take values of w, but more often the cases of Figure 2a–c occur.

Based on the above, we propose an adaptive heuristic function applicable to control scalar risk and vector risk in ship track planning, i.e., the following Equation (10).

$$\begin{aligned} \widetilde{h}(cell_i, g(cell_i)) &= h(cell_i) \cdot g(cell_i) \\ &= euclidean(cell_i, \text{goalpoint}) \cdot g(cell_i) \\ &= g(x_i, y_i)\sqrt{(x_i - x_g)^2 + (y_i - y_g)^2}, \end{aligned} \tag{10}$$

where $\widetilde{h}(\cdot)$ is the adaptive heuristic function, which requires independent variables $cell_i$ and $g(cell_i)$. $euclidean(cell_i, \text{goalpoint})$ denotes the Euclidean distance from the grid $cell_i$ to the

endpoint. By changing the heuristic function of A* to $\widetilde{h}(\cdot)$, we obtain the results in Figure 2d. Compared with Figure 2a–d, it is obviously closer to the target of $h(cell_i) = h*(cell_i)$. Why does it achieve this effect? First, it expands each grid while modulating the heuristic function values according to $cell_i$ and $g(cell_i)$. This ensures its flexibility. Second, its unique tuning mechanism. Due to $g(cell_i)$, $euclidean(cell_i, \text{goalpoint})$ can easily reach an order of magnitude with $g(cell_i)$. It is a prerequisite for maximizing the performance of A*, so now how to make $\widetilde{h}(cell_i) = h*(cell_i)$. In discussing Figure 2c, we found that its first half of the journey is too much $h(cell_i)$ than $h*(cell_i)$ because w is too large, so it does not expand the necessary number of grids. But the adaptive heuristic function need not be a concern. The value of $euclidean(cell_i, \text{goalpoint})$ is larger and $g(cell_i)$ is smaller in the first half of the journey; the value of $euclidean(cell_i, \text{goalpoint})$ is smaller and $g(cell_i)$ is larger in the second half of the journey. Then $\widetilde{h}(\cdot)$ satisfies the variation law of $h*(cell_i)$. And because $\widetilde{h}(\cdot)$ is acted by $g(cell_i)$ to the order of magnitude of $h*(cell_i)$. So $\widetilde{h}(\cdot)$ has the property of being closer to $h(\cdot)$ than the traditional $h*(cell_i)$. Satisfying this property, A* will expand the grid appropriately, thus enhancing its speed and effectiveness.

In conclusion, when A* is given a heuristic function that is suitable for the task scenario, it will significantly reduce the useless grid expansion and enhance the risk control capability of A*. Based on the adaptive heuristic function, we propose a segmented parallel A* for route planning, inspired by the efficient data processing model of "allocate and run simultaneously" in parallel computing. The specific steps are as follows:

Step 1: Generate a standard path from the start point to the end point, denoted as $path_o$. The default standard path is a terrain risk minimization path generated by A* loaded with adaptive heuristic functions. However, experimentally, any user-approved path from the origin to the destination (empirical path) is allowed by SPA*, which enables further detailed planning of the standard path. This paper provides one of the following ways to generate terrain risk data:

(a). Collect the ranks and coordinates of all navigable grids and non-navigable grids (islands or land) to form a matrix.

(b). Calculate the Euclidean distance of each navigable grid to all non-navigable grids in turn. If there are m navigable grids and n non-navigable grids, then there are mn distance values. If the row is the navigable grid number and the row is the non-navigable grid number, then a $m \times n$-dimensional matrix can be formed.

(c). Find the minimum value in each row and then delete the rest of the values in each row, leaving a $m \times 1$-dimensional vector. This vector retains the distance values of the nearest non-navigable grid for each navigable grid.

(d). Normalize the $m \times 1$-dimensional vector in (c) by the maximum and minimum values. The values in the vector are then inverted. The inverse processing is performed using the circular arc function (Equation (11)). It simultaneously slows down its increase to 1 and makes the terrain risk data more reasonable.

$$\begin{aligned} f_M &= 1 - \sqrt{1 - (z-1)^2}, \\ & z = [z_1, z_2, \cdots, z_m]^T,\ 0 \le z_i \le 1,\ i = 1, 2, \cdots, m, \end{aligned} \tag{11}$$

where z represents the risk value that needs reverse processing; f_M represents terrain risk data. Applying the above (a)~(d) to the illustrated sea area, the topographic risk heat map shown in Figure 3 can be obtained. The redder the color, the greater the terrain risk, and the bluer the color, the smaller the terrain risk. In summary, the processed vector can quantitatively reflect the risk of reef cuts and groundings of vessels passing through the grid. The smaller the value of a row in the vector, the lower the terrain risk of the corresponding grid; conversely, the higher the terrain risk.

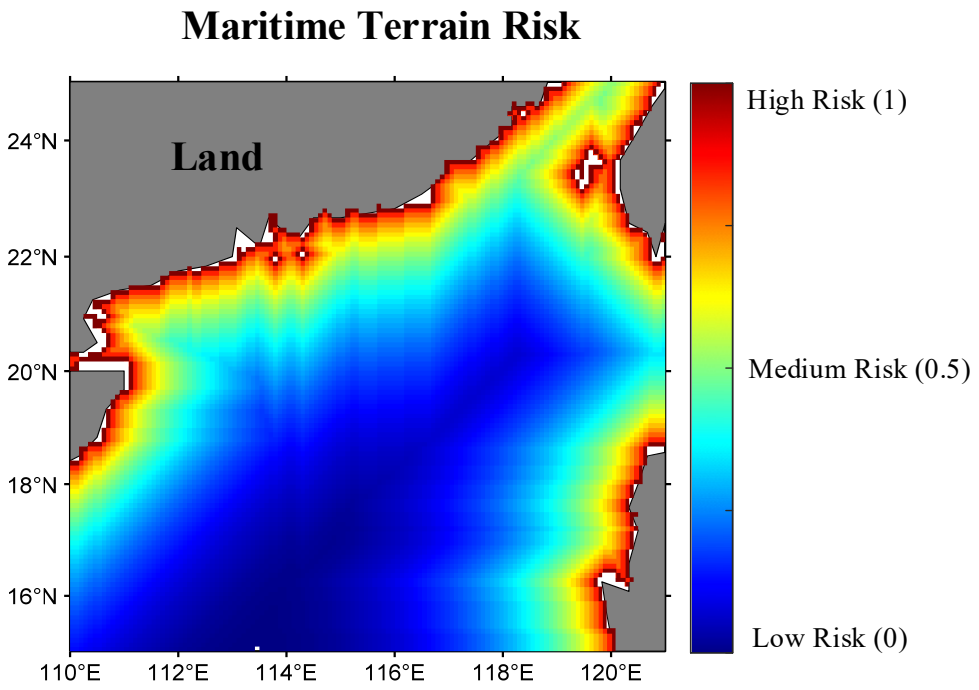

Figure 3. Heat map on terrain risk.

Step 2: Next, the standard paths are divided equally. The small segments obtained from the equalization of the standard paths are assigned to run A* on the CPU cores separately by a control algorithm. The control algorithm for the tasks on the CPU is already packaged in Python, C++, Java, and MATLAB. In this paper, we choose MATLAB software and use the "parfor" command to decompose the original serial cycle into multiple parallel cycles.

Step 3: Each small segment of the planned trajectory is stitched together to form a risk-minimized trajectory from origin to destination. For long-distance planning tasks, multiple constraints, and high-resolution data requirements, SPA* dramatically reduces computational time consumption while maintaining performance through parallel allocation. It is more compatible than some current parallel path planning algorithms. SPA* is easy to adapt and straightforward to use for both risk-minimization and shortest-circuit problems. See Algorithm 1.

Algorithm 1 SPA*

Input: F_1; F_2; $path_o$; m
// F_1 and F_2 are 3D arrays, with the first two dimensions being longitude and latitude, and the third dimension being time. $F_1(i)$ and $F_2(i)$ represents the risk data at time period i. m is the number of segments.
Output: $path_{SPA*}$
$S \leftarrow$round(linspace($path_o$, m))
// "round()" represents rounding all elements to the nearest whole number.
// "linspace(a, b)" represents dividing vector a into b segments.
// S is the $m \times 2$ array. Each row stores the start and end indexes of each segment of $path_o$.
parfor $i \leftarrow 1$ **to** m **do**
$par_path(i) \leftarrow \mathrm{A}^*_{ahf}(S(i,1), S(i,2), F_1(i), F_2(i))$
// "$\mathrm{A}^*_{ahf}(s, e, F_1, F_2)$" represents using A * with an adaptive heuristic function for path planning, starting from s and ending at e, where the risk data used are F_1 and F_2.
end
$path_{SPA*} \leftarrow [par_path(1), par_path(2), \cdots, par_path(m)]$
return $path_{SPA*}$

Figure 4 depicts the advantages of SPA* when performing dynamic path planning. Conventional path planning requires a full route from the current position to the end point for each time period. Then only the part of the route that can be reached by the ship within the specified time period is taken, and the remaining routes are deleted. If the dynamic planning from the starting point to the end point is divided into four time periods, then three long and short redundant paths will appear at the end. They significantly increase the algorithm's running time. In combination with the algorithm pseudo-code, SPA* is able to assign routes to different time periods and plan them simultaneously. This results in zero redundant routes, so the running time of SPA* is very fast.

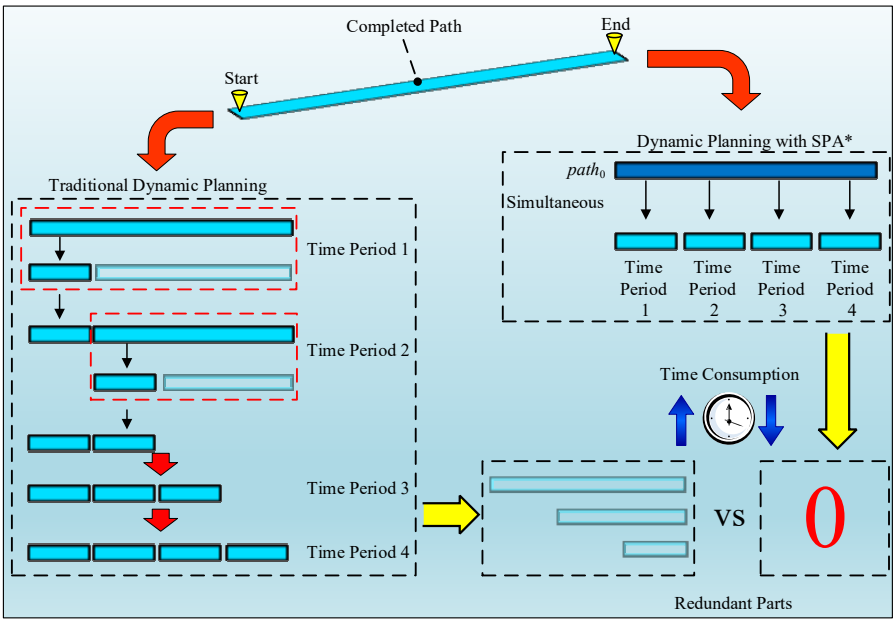

Figure 4. For dynamic planning, the implementation of SPA* is compared with the traditional implementation.

3. Experiments

*3.1. Static Route Planning Experiments by SPA**

In this section, we use the traditional path planning algorithm and the swarm intelligence (SI)-based path planning algorithm as a comparison of SPA*. Traditional path planning algorithms include Dijkstra, A*, and BA*; SI-based path planning algorithms include ACO, HHO, and SSA. The static route path planning experiments refer to route planning from the starting point to the end point by fixing the risk assessment data over a certain period of time. The main observation indexes on a route are the running time of algorithms, the average assessment value of wind and wave risk f_1, the average assessment value of comprehensive risk f_2, and the rate of violating IMO (R-VIMO). Then R-VIMO is calculated using Equation (12).

$$\begin{aligned} \mathrm{R-VIMO}(Path) &= \frac{\sum_{i=2}^{n} I(q_i, v_i, T_E(i))}{n}, \\ I(q_i, v_i, T_e(i)) &= \begin{cases} 1, \text{ if } q_i, v_i \text{ and } T_E(i) \text{ satisfy Constraint I or II} \\ 0, \text{ otherwise} \end{cases} \end{aligned} \tag{12}$$

where R-VIMO(*Path*) represents R-VIMO of *Path*; q_i is the angle between the wave direction and the ship's heading when the ship travels to the ith grid according to the planned path *Path*; v_i is the actual velocity when the ship travels to the ith grid according to the planned path *Path*; $T_E(i)$ is the encountered period with waves when the ship travels to the ith grid according to the planned path *Path*; q_i, v_i and $T_E(i)$ are from *Path*. $I(q_i, v_i, T_e(i))$ is an indicator function that determines whether three variables satisfy Constraints I and II.

The smaller the running time of algorithms f_1, f_2, and R-VIMO, the better the performance of the algorithm. R-VIMO is an important standard. If the R-VIMO of a route is greater than 0, it means that it has a section that does not meet the IMO standard. Keeping R-VIMO at 0 is a test of the performance of the planning algorithm. This paper argues that the path of R-VIMO = 0 is absolutely better than that of R-VIMO > 0. There are some red segments in the paths marked by different colors in Figure 5 and the last figure in paper. The red segments indicate violations of Constraint I or Constraint II in Section 2.1.

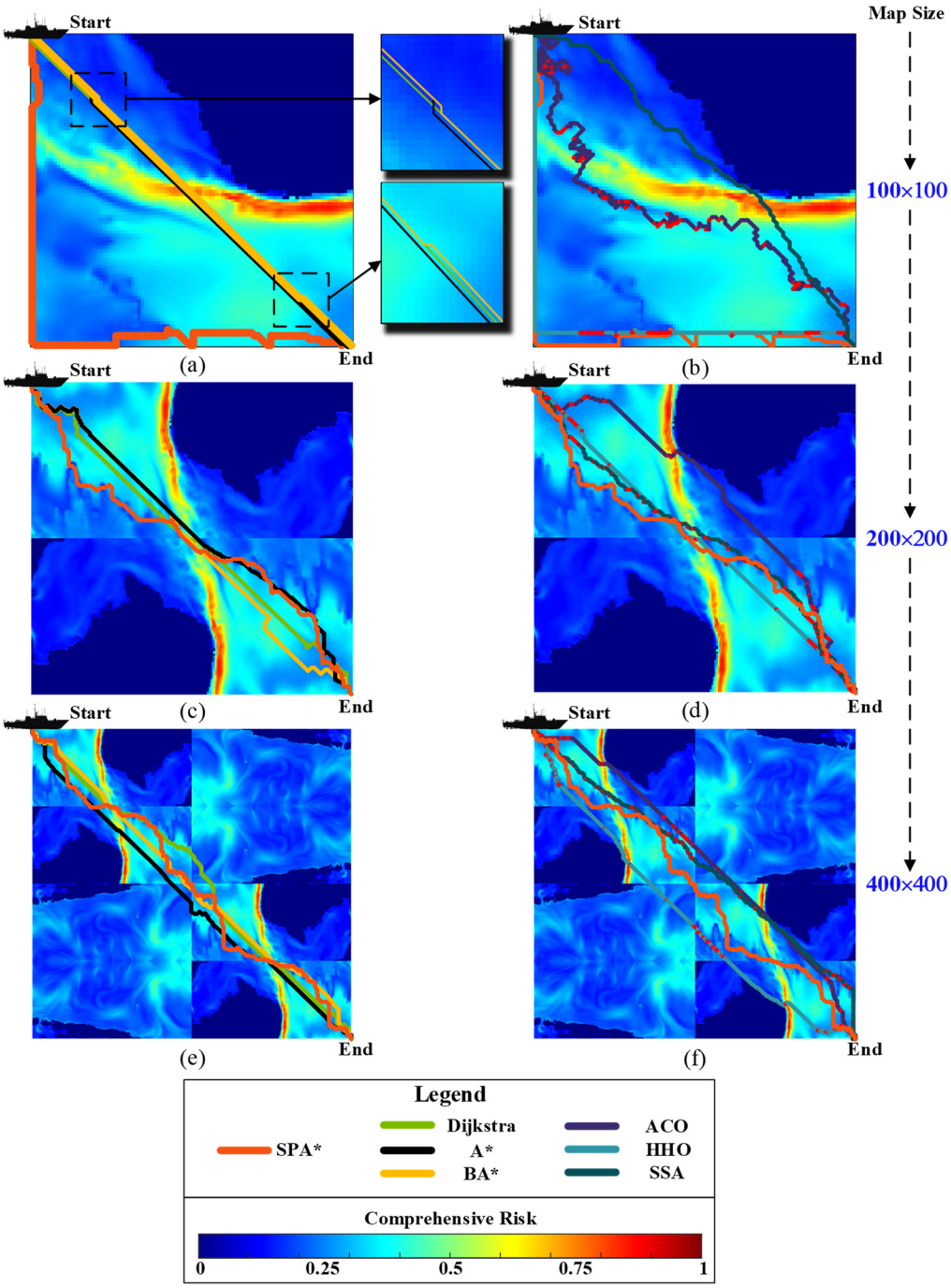

Figure 5. Static planning routes of SPA*, traditional path planning and SI-based path planning for different map sizes. (**a**,**c**,**e**) are the routes planned by SPA* and traditional path planning algorithms for the corresponding map sizes; (**b**,**d**,**f**) are the routes planned by SPA* and SI-based path planning algorithms for the corresponding map sizes.

The weights in the cost function of Dijkstra, A *, and BA * are consistent with SPA*. The w of the heuristic function in A * and BA * is 0.15. The maximum number of iterations in ACO is 400, the initial population is 50, α is 2, β is 5, ρ is 0.3, and Q is 200. The maximum number of iterations of HHO is 400, and the population size is 50. The maximum number of iterations of SSA is 400, the initial population is 50, the warning value is 0.6, and the proportion of discoverers and participants is 0.7 and 0.3, respectively. The number of segments m in SPA* is 7, where the heuristic function of A* is the adaptive heuristic function proposed in this paper. The starting point of each experiment is set at the top left corner of the map, and the end point is set at the bottom right corner of the map.

The cruising speed of the ship in the experiment is 30 knots per hour, and the displacement is 54,500 tons. And T_r is 10.0065 s. The length, width, and height of the ship are 306.4, 74.4, and 14 meters, respectively. According to the main scale parameters of the ship, the coefficients a_1~a_4 in Equation (2) are taken as 1.08, 0.126, 0.00277, and 2.33×10^{-7}, respectively [15,16,22,23,38]. The experimental risk assessment data were taken from the sea area of latitude -15° to 15 and longitude 100° to 135°. It includes vector data of wind and waves, wave height, wave speed, wave period, sea surface temperature, and sea surface salinity. The time range of the data is 2022-11-1~2022-11-7, and the four time periods recorded each day are 00:00, 06:00, 12:00 and 18:00; spatial resolution: $1/12^\circ$; data scale: $421 \times 361 \times 28$ (longitude-latitude-time). We averaged the data on the first day, and the regions with distinct risk characteristics were constituted into the data sets of 100×100, 200×200 and 400×400 in Figure 5. Each of the above cases was experimented 100 times, and the average value of each indicator obtained was the average value.

The relevant experimental results are organized in Table 2. Under different map sizes, the R-VIMO of SPA * and traditional path planning algorithms are all 0. In SI-based path planning, only SSA's R-VIMO is 0 at 100×100. Compared with other algorithms, SPA * shows obvious advantages in four indicators. Especially for time, even in the size of 400×400, SPA * still maintains the calculation speed of completing a planning within 1.5 s. In contrast, A * and BA *, which are faster in traditional path planning, need 7.0562 and 22.4825 s to complete a planning, respectively. The fastest SSA in SI-based path planning still needs 13.0777 s to complete a planning. However, the R-VIMO of SSA is 0~27.47%, which means that its planning performance has serious problems and cannot meet the IMO standard.

Table 2. Static planning results of SPA*, traditional path planning, and SI-based path planning for different map sizes.

Indicator	Our Algorithm	Traditional Path Planning			SI-Based Path Planning			Map Size
	SPA*	Dijkstra	A*	BA*	ACO	HHO	SSA	
Time(s)	0.0312	0.8908	0.5708	0.4670	202.4345	328.2276	2.5359	100×100
f_1	0.1051	0.1404	0.1395	0.1404	0.1254	0.1019	0.1238	
f_2	0.2833	0.3230	0.3228	0.3224	0.2896	0.2812	0.2558	
R-VIMO	0	0	0	0	16.94%	14.72%	0	
Time(s)	0.0805	6.9234	1.5625	0.7388	1000.2133	923.3016	6.3452	200×200
f_1	0.1154	0.1287	0.1250	0.1315	0.1307	0.1343	0.1170	
f_2	0.2763	0.3079	0.3046	0.3227	0.2525	0.3144	0.3017	
R-VIMO	0	0	0	0	18.47%	22.38%	27.47%	
Time(s)	1.2354	159.3826	7.0562	22.4825	1723.6522	1523.3016	13.0777	400×400
f_1	0.1145	0.1285	0.1309	0.1288	0.1333	0.1355	0.1172	
f_2	0.2754	0.3008	0.3024	0.3090	0.2575	0.2495	0.2824	
R-VIMO	0	0	0	0	16.29%	15.07%	17.37%	

Combined with Figure 5, it is observed that the routes obtained by SPA* are better suited to avoid the high-risk area marked in red. At 100 × 100, SPA* selects a detour near 90 degrees to avoid the risk zone; in traditional path planning, the algorithm is straight through the risk area; in the SI-based path planning algorithm, ACO and SSA fall into two local optimal solutions, and the path finally crosses the risk area. ACO passes through the medium-risk area, and SSA passes through the high-risk area. At 200 × 200, SPA* not only avoids the high-risk area from the middle but also bypasses the medium-risk area of the path. The algorithms in traditional path planning do not bypass the medium-risk area of the path; in the SI-based path planning algorithm, ACO falls into the local optimal solution and crosses the high-risk area. At 400 × 400, SPA* also identifies and avoids risk areas. The algorithms in traditional path planning and SI-based path planning began to show serious performance deficiencies. Most of the paths in them appear as straight lines without planning purpose, just like the shortest path without considering risk. In particular, all the planned routes in SI-based path planning have a route very close to the island, which undoubtedly increases the risk of ship grounding.

In general, the time of SPA* is reduced by about 5~12,425 times compared to each algorithm, and the R-VIMO of it is always maintained at 0. At the same time, in the control of f_1 and f_2, the performance of SPA* is improved by 7.68%~25.14% and 8.44%~14.38%, respectively, compared with each algorithm. According to the experiments in this section, SPA * shows excellent and fast risk planning ability in static navigation path planning.

3.2. Parametric Experiments for the Number of Segments m

SPA* shows excellent performance in static navigation path planning experiments, but the number of segments m of SPA * is only set to 7. If the m of SPA * is changed, will the performance of SPA * be greatly affected? Or how to set the m of SPA* to make SPA* show better performance?

In order to obtain answers to the above questions, we designed an adequate parametric experiment on m for SPA*. It is based on Section 3.1, adjusting m for different map size cases: 6~70 for 100 × 100, 6~100 for 200 × 200, and 6~200 for 400 × 400. And the changes in the running times of the algorithms f_1 and f_2 are observed (the R-VIMO of SPA* is always 0). We also added an additional indicator, namely the same ratio as the generated path (SR-GP). SR-GP is calculated by the following Equation (13).

$$\frac{ismember(Path_{SPA*}, Path_{gp})}{length(Path_{SPA*})}, \tag{13}$$

where $ismember(a, b)$ represents the number of the same components of vectors a and b; $length(\cdot)$ is a function that counts the length of the vectors inside the brackets. A larger SR-GP means a higher degree of identity with the generated path. SR-GP equal to 1 means that the generated path has no new information after SPA*. Hence, we want the SR-GP to be as small as possible. In this way, the path is expected to have better risk-control ability.

The experimental results are shown in Figure 6. The x and y axes of Figure 6a,d,g indicate the SR-GP and the running time, respectively, and the size of the bubble indicates the number of segments m. The blue markers on Figure 6b,e,h indicate f_1; the yellow markers on Figure 6c,f,i indicate f_2; where 100 × 100 is represented by a circular marker; 200 × 200 is represented by a diamond marker; and 400 × 400 is represented by a triangular marker.

For 100 × 100, the SPA* algorithm time and SR-GP both show an increasing trend as the number of segments increases (the yellow curve is the trend line). And at this time, both f_1 and f_2 show a tendency to rise first and then stabilize around a certain value to fluctuate, and the threshold at which they tend to stabilize is around m = 40. Therefore, we believe that in the case of 100 × 100, the lower the number of segments, the better, but not less than 6. For 200 × 200, as the number of segments increases, the SR-GP of SPA* continues to rise, while the overall SPA* algorithm running time tends to fluctuate downward. And at this time, both f_1 and f_2 show the trend of first rising and then stabilizing around a certain value fluctuating, and the threshold at which they tend to stabilize is around m = 30. So

we think that m that makes SR-GP maintain between 0.2 and 0.3 is preferable in the case of 200 × 200. The range of m satisfying the above condition is 10~20. For 400 × 400, the SR-GP of SPA* continues to rise as the number of segments increases, while the running time of SPA* first decreases rapidly and then the rate of decrease tends to level off. Finally, the running time of SPA* stabilizes at a certain value. And at this time, both f_1 and f_2 show a trend of first rising and then stabilizing around a certain value fluctuating, and the threshold at which they tend to stabilize is around m = 60. So, we think that m that makes SR-GP maintain between 0.1 and 0.2 is preferable in the case of 400 × 400. The range of m satisfying the above condition is 15~30.

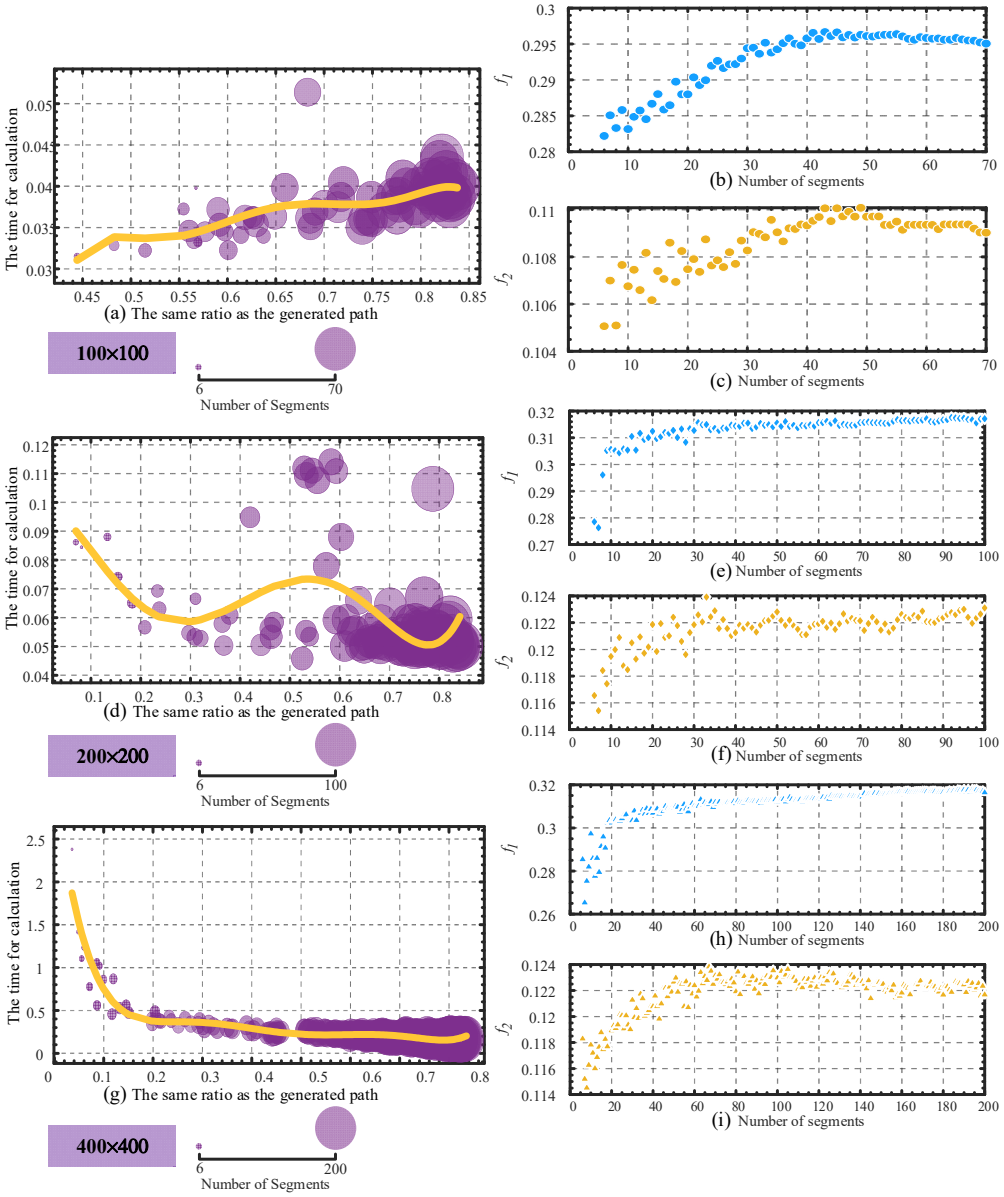

Figure 6. Experiments on the sensitivity of m for different map sizes. (**a**), (**d**), and (**g**) are bubble plots for different map sizes with SR-GP, the time for calculation and the number of segments as their x-axis, y-axis, and bubble size, respectively; (**b**), (**e**), and (**h**) are scatter plots for different map sizes with the number of segments and f_1 as their x-axis and y-axis, respectively; (**c**), (**f**), and (**i**) are scatter plots for different map sizes with the number of segments and f_2 as their x-axis and y-axis, respectively.

Additionally, note that SPA* does not have a large range of runtime changes when the number of segments m is adjusted significantly. For 100 × 100, the running time change is about 0.01 s; for 200 × 200, the running time change is about 0.04 s; for 400 × 400, the running time change is about 1.8 s. Conversely, observe that the SPA* on f_1 and f_2 are more variable and are more important indicators in route planning. Therefore, we need to consider the value of the number of segments m in different cases, so that the performance of SPA* is guaranteed.

Based on the above analysis, the optimal intervals of m under different map sizes are given. For 100 × 100, the lower the number of segments, the better, but not less than 6; for 200 × 200, the number of segments should be 10~20; for 400 × 400, the number of segments should be 15~30. The results show that the optimal performance is near 15 grids per segment for various map sizes. After limiting the range of m, SPA * also shows strong control ability on f_1 and f_2. Therefore, we believe that SPA * has the advantages of fewer parameters, strong compatibility, and robustness.

3.3. Dynamic Route Planning Experiments by SPA*

Dynamic route planning experiments are rolling route planning from start to finish for multiple time periods or multiple days of data at a given forecast size. In general, the planning algorithm first performs a static planning from the starting point to the end point. Then it determines the starting point for the next time period based on the distance that can be traveled each day, and then performs another static planning for reaching the end point. And so on, until the above static planning is done once for the last time period. Combined with the above-mentioned parameter experiments, the number of segments m in SPA* can be taken as the corresponding value to the total forecast data size. When m takes the corresponding value, SPA* spreads the whole planning task to the corresponding time slots for simultaneous computation, i.e., parallel computation. For example, when the total forecast data size is 7 days or 28 time periods, m can be taken as 7 or 28. The start point and end point are shown in Figure 7. SPA * determines the daily driving distance according to the number of segments, and the remaining algorithms sail 10 grids per day.

The results of the dynamic experiments are shown in Table 3. SPA* takes only about 9 s to complete a dynamic experiment when the scale of the pre-reported data is Daily. The fastest algorithm is SSA, which takes about 45 s to complete a dynamic experiment, and the slowest is Dijkstra, which takes about 900 s to complete a dynamic experiment. The time required for SPA* is even less, about 2 s, when the scale of the forecast data is "Every 6 h". However, at this time, the risk control capability of SPA* decreases a bit compared to the "Daily" case. This is in line with the results of the parametric experiments. The fastest of the other algorithms, SSA, takes 321 s to complete a planning, which is nearly 5.5 min. The R-VIMO of both SSA and BA* is greater than 0. This indicates that the planning capability of these algorithms is not sufficient to meet the corresponding constraints with the current data volume.

Combining Figure 7, it is observed that the routes obtained by SPA* are relatively well avoiding the high-risk areas marked in red. First of all, we will examine the case where the scale of the forecast data scale is "Daily". When running to the first day, as shown in Figure 7b, SPA* crosses the striped risk area from the lower risk area. The SSA crosses the striped risk area closer to the middle and high-risk areas; when the run reaches the seventh day, as shown in Figure 7c, SPA* bypasses a low-risk gap at the lower left of the circular risk area to reach the end point. While all other algorithms crossed the circular risk area. The next observation is the forecast data size of "Every 6 h". When running to the 3rd time period, as in Figure 7c, SPA* continues to maintain its performance. When running to the 26th time period, SPA* also chooses to cross the circular risk zone, but at a different location than the other algorithms. Combined with the values of f_1 and f_2 of SPA* in Table 3, we can affirm that the planning carried out by SPA* has a better ability to control risk compared with other algorithms.

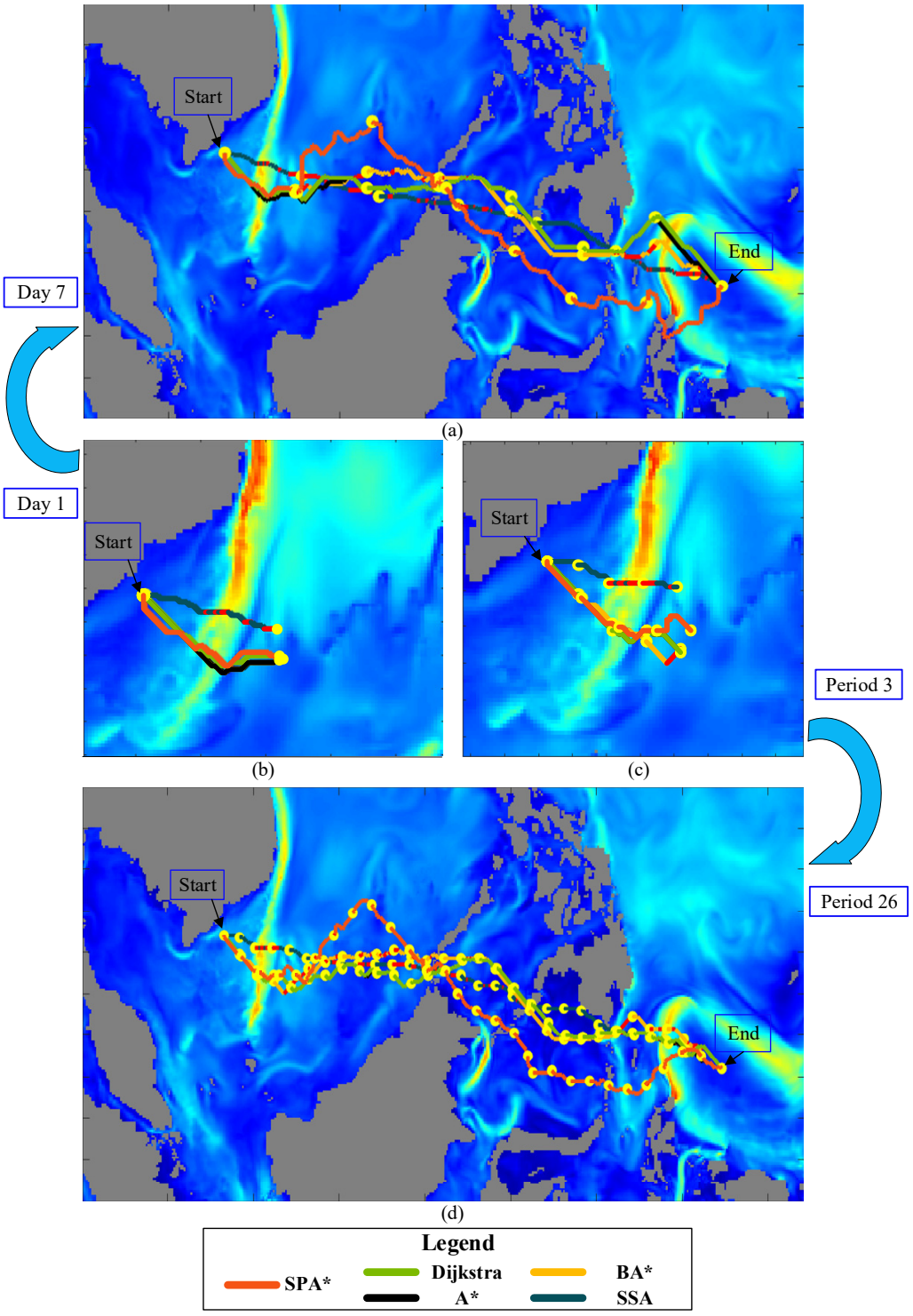

Figure 7. Dynamic planning routes of SPA*, traditional path planning and SI-based path planning under different data scales. (**a**,**b**) are the routes planned by the algorithms on day 7 and day 1, respectively, with the forecast data scale of "Daily"; (**c**,**d**) are the routes planned by the algorithms in period 3 and period 26, respectively, with the forecast data scale of "Every 6 h".

Table 3. Dynamic planning results for SPA*, traditional path planning, and SI-based path planning with different data scales.

Indicator	Our Algorithm	Traditional Path Planning			SI-Based Path Planning	Forecast Data Scale
	SPA*	Dijkstra	A*	BA*	SSA	
Time(s)	9.3600	904.8657	269.0035	84.8583	45.3768	Daily (Total 7 days)
f_1	0.1166	0.1228	0.1238	0.1213	0.1082	
f_2	0.2137	0.2384	0.2374	0.2432	0.2420	
R-VIMO	0	0	0	16.78%	31.60%	
Time(s)	2.4335	3073.3542	937.1178	327.9520	321.9712	Every 6 h (total 28 periods)
f_1	0.1137	0.1256	0.1254	0.1235	0.1084	
f_2	0.2265	0.2441	0.2449	0.2534	0.2462	
R-VIMO	0	0	0	22.87%	30.03%	

Experiments show that the time of SPA* is reduced by about 4.8~1262.9 times compared with each algorithm, and the R-VIMO is always 0. At the same time, in the control of f_1 and f_2, the performance of SPA* is improved by 3.87%~9.47% and 7.21%~10.36%, respectively, compared with other algorithms. All in all, SPA * shows excellent and fast risk planning ability in dynamic route path planning.

4. Conclusions

To overcome the problems of long computational time consumption, severe performance degradation, and poor compatibility of traditional maritime route planning in terms of long-distance planning tasks, multiple constraints, and high-resolution data nowadays. In this paper, a robust, high-performance and computationally efficient parallel route planning algorithm, namely SPA*, is proposed. The theory of parallel computing is applied in SPA*. SPA * is suitable for solving the minimum risk path problem, and risk values are obtained by evaluating predictive data for dynamic, uncertain marine environments. Through static planning experiments with three map sizes and dynamic planning experiments with two sizes of forecast data in Indonesian waters, we obtain the following conclusions:

(1) The performance and computational efficiency of SPA* is excellent. In static experiments, the comparison subjects are traditional path planning algorithms (Dijkstra, A*, and BA*) and SI-based path planning algorithms (ACO, HHO, and SSA). The time of SPA* is reduced by about 5~12,425 times compared with each algorithm, and the R-VIMO of it is always kept at 0. Meanwhile, in the control of f_1 and f_2, the performance of SPA* is improved by 7.68%~25.14% and 8.44%~14.38% compared with each algorithm. In dynamic experiments, SPA* is compared with traditional path planning algorithms (Dijkstra, A*, and BA*) and SI-based path planning algorithms (SSA). The time of SPA* is reduced by about 4.8~1262.9 times compared to each algorithm, and the R-VIMO of it is always kept at 0. In the control of f_1 and f_2, the performance of SPA* is improved by 3.87%~9.47% and 7.21%~10.36%, respectively, compared with each algorithm.

(2) SPA* has the advantage of fewer parameters and strong robustness. In the experiments for the only parameter "the number of segment m", it is adjusted substantially, but the time of SPA* is changed to a small range. For 100×100, 200×200, and 400×400, the change in running time is about 0.01 s, 0.04 s, and 1.8 s, respectively. For the control of risk indicators f_1 and f_2, we analytically give some reasonable intervals of m for SPA* at different map sizes. For 100×100, the lower m the better, but not less than 6; for 200×200, m should be 10~20; for 400×400, m should be 15~30. After limiting the range of m, SPA* shows great control at f_1 and f_2.

When the planning scenario satisfies long-distance planning tasks, multi-constraint conditions, and high-resolution data requirements, we note that our algorithm outperforms the swarm intelligence algorithm. This is likely because the traditional metaheuristic procedures to solve the path planning problem step is: First search for the key points of a path, then connect the key points to form a path, and finally iteratively obtain the planning path. Then, the connection algorithm determines, to a large extent, the sensitivity of the planning path to grid data. If the connection algorithm is very rough, then the planning path will take a long time and the effect will be poor. And our algorithm is a variant of A* that plans for every point on the path. It becomes more competitive when the heuristic function of A* is optimized and a parallel structure is designed for A*.

The parallel computational structure designed in SPA* is of general interest and may have promising applications in other deterministic path-planning algorithms as well. In summary, we believe that SPA* is a representative work in the field of maritime route planning that provides a high-performance and robust route planning scheme based on parallel computing and the adaptive heuristic function. It takes very little time and has a stable performance that can be the basis for adding several other optimized mechanisms, such as speed planning and *k*-shortest route planning, for the commander to choose from. However, it should be noted that: (1) SPA* has been designed with a parallel structure, which has relatively high computer hardware; (2) SPA* requires a standard path to run, which is obtained in this paper by considering terrain risk and distance. There is still room for optimization for generating the standard path, and researching an excellent standard path generation algorithm will improve SPA* performance at another level.

Author Contributions: Conceptualization, H.L. and L.Q.; methodology, H.L. and L.Q.; software, H.L.; validation, H.L.; investigation, H.L. and Z.G.; writing—original draft preparation, H.L.; writing—review and editing, L.Q.; supervision, M.H. and X.W.; funding acquisition, H.L., L.Q. and M.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China (Grant Nos. 42375016 and 51609254), the Natural Science Foundation of Hunan Province (2023JJ10054), the NUPTSF (Grant Nos. NY219161 and NY220035) and the Postgraduate Research & Practice Innovation Program of Jiangsu Province (Grant No. SJCX23_0249).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Gourvenec, S.; Sturt, F.; Reid, E.; Trigos, F. Global assessment of historical, current and forecast ocean energy infrastructure: Implications for marine space planning, sustainable design and end-of-engineered-life management. *Renew. Sust. Energ. Rev.* **2022**, *154*, 111794. [CrossRef]
2. Kuroda, M.; Sugimoto, Y. Evaluation of ship performance in terms of shipping route and weather condition. *Ocean Eng.* **2022**, *254*, 111335. [CrossRef]
3. Li, M.; Luo, H.; Zhou, S.; Kumar, G.M.S.; Guo, X.; Law, T.C.; Cao, S. State-of-the-art review of the flexibility and feasibility of emerging offshore and coastal ocean energy technologies in East and Southeast Asia. *Renew. Sust. Energ. Rev.* **2022**, *162*, 112404. [CrossRef]
4. Shiiba, N.; Wu, H.H.; Huang, M.C.; Tanaka, H. How blue financing can sustain ocean conservation and development: A proposed conceptual framework for blue financing mechanism. *Mar. Policy* **2022**, *139*, 104575. [CrossRef]
5. Zhu, H.; Wang, Y.; Li, X. UCAV path planning for avoiding obstacles using cooperative co-evolution Spider Monkey Optimization. *Knowl.-Based Syst.* **2022**, *246*, 108713. [CrossRef]
6. Zhang, X.; Xia, S.; Li, X.; Zhang, T. Multi-objective particle swarm optimization with multi-mode collaboration based on reinforcement learning for path planning of unmanned air vehicles. *Knowl.-Based Syst.* **2022**, *250*, 109075. [CrossRef]
7. Li, H.; Zhao, T.; Dian, S. Forward search optimization and subgoal-based hybrid path planning to shorten and smooth global path for mobile robots. *Knowl.-Based Syst.* **2022**, *258*, 110034. [CrossRef]

8. Cummings, M.L.; Buchin, M.; Carrigan, G.; Donmez, B. Supporting intelligent and trustworthy maritime path planning decisions. *Int. J. Hum.-Comput. Stud.* **2010**, *68*, 616–626. [CrossRef]
9. Li, M.; Xie, C.; Li, X.; Karoonsoontawong, A.; Ge, Y.-E. Robust liner ship routing and scheduling schemes under uncertain weather and ocean conditions. *Transp. Res. Part C Emerg. Technol.* **2022**, *137*, 103593. [CrossRef]
10. Sung, I.; Zografakis, H.; Nielsen, P. Multi-lateral ocean voyage optimization for cargo vessels as a decarbonization method. *Transp. Res. Part D Transp. Environ.* **2022**, *110*, 103407. [CrossRef]
11. Xi, M.; Yang, J.; Wen, J.; Liu, H.; Li, Y.; Song, H.H. Comprehensive ocean information-enabled AUV path planning via reinforcement learning. *IEEE Internet Things* **2022**, *9*, 17440–17451. [CrossRef]
12. Zhao, W.; Wang, H.; Geng, J.; Hu, W.; Zhang, Z.; Zhang, G. Multi-objective weather routing algorithm for ships based on hybrid particle swarm optimization. *J. Ocean. Univ. China* **2022**, *21*, 28–38. [CrossRef]
13. Singh, Y.; Sharma, S.; Sutton, R.; Hatton, D.; Khan, A. A constrained A* approach towards optimal path planning for an unmanned surface vehicle in a maritime environment containing dynamic obstacles and ocean currents. *Ocean Eng.* **2018**, *169*, 187–201. [CrossRef]
14. Fabbri, T.; Vicen-Bueno, R. Weather-routing system based on METOC navigation risk assessment. *J. Mar. Sci. Eng.* **2019**, *7*, 127. [CrossRef]
15. Grifoll, M.; Borén, C.; Castells-Sanabra, M. A comprehensive ship weather routing system using CMEMS products and A* algorithm. *Ocean Eng.* **2022**, *255*, 111427. [CrossRef]
16. Shin, Y.W.; Abebe, M.; Noh, Y.; Lee, S.; Lee, I.; Kim, D.; Bae, J.; Kim, K.C. Near-optimal weather routing by using improved A* algorithm. *Appl. Sci.* **2020**, *10*, 6010. [CrossRef]
17. Yang, Z.; Yu, X.; Dedman, S.; Rosso, M.; Zhu, J.; Yang, J.; Xia, Y.; Tian, Y.; Zhang, G.; Wang, J. UAV remote sensing applications in marine monitoring: Knowledge visualization and review. *Sci. Total Environ.* **2022**, *838*, 155939. [CrossRef]
18. Yuan, S.; Li, Y.; Bao, F.; Xu, H.; Yang, Y.; Yan, Q.; Zhong, S.; Yin, H.; Xu, J.; Huang, Z. Marine environmental monitoring with unmanned vehicle platforms: Present applications and future prospects. *Sci. Total Environ.* **2023**, *858*, 159741. [CrossRef]
19. Zhang, S.; Mei, Y.; Xia, T.; Cao, Z.; Liu, Z.; Li, Z. Simultaneous measurement of temperature and pressure based on Fabry-Perot Interferometry for marine monitoring. *Sensors* **2022**, *22*, 4979. [CrossRef]
20. Lee, T.; Kim, H.; Chung, H.; Bang, Y.; Myung, H. Energy efficient path planning for a marine surface vehicle considering heading angle. *Ocean Eng.* **2015**, *107*, 118–131. [CrossRef]
21. Ma, W.; Han, Y.; Tang, H.; Ma, D.; Zheng, H.; Zhang, Y. Ship route planning based on intelligent mapping swarm optimization. *Comput. Ind. Eng.* **2023**, *176*, 108920. [CrossRef]
22. Chen, Y.-Q.; Guo, J.-L.; Yang, H.; Wang, Z.-Q.; Liu, H.-L. Research on navigation of bidirectional A* algorithm based on ant colony algorithm. *J. Supercomput.* **2021**, *77*, 1958–1975. [CrossRef]
23. Liu, L.; Wang, B.; Xu, H. Research on Path-Planning Algorithm Integrating Optimization A-Star Algorithm and Artificial Potential Field Method. *Electronics* **2022**, *11*, 3660. [CrossRef]
24. Khedr, A.M.; Al Aghbari, Z.; Khalifa, B.E. Fuzzy-based multi-layered clustering and ACO-based multiple mobile sinks path planning for optimal coverage in WSNs. *IEEE Sens. J.* **2022**, *22*, 7277–7287. [CrossRef]
25. Saeed, R.A.; Omri, M.; Abdel-Khalek, S.; Ali, E.S.; Alotaibi, M.F. Optimal path planning for drones based on swarm intelligence algorithm. *Neural Comput. Appl.* **2022**, *34*, 10133–10155. [CrossRef]
26. Alweshah, M.; Almiani, M.; Almansour, N.; Al Khalaileh, S.; Aldabbas, H.; Alomoush, W.; Alshareef, A. Vehicle routing problems based on Harris Hawks optimization. *J. Big Data* **2022**, *9*, 42. [CrossRef]
27. Zhang, R.; Li, S.; Ding, Y.; Qin, X.; Xia, Q. UAV Path Planning Algorithm Based on Improved Harris Hawks Optimization. *Sensors* **2022**, *22*, 5232. [CrossRef]
28. Gharehchopogh, F.S.; Namazi, M.; Ebrahimi, L.; Abdollahzadeh, B. Advances in sparrow search algorithm: A comprehensive survey. *Arch. Comput. Methods Eng.* **2023**, *30*, 427–455. [CrossRef]
29. Zhang, Z.; He, R.; Yang, K. A bioinspired path planning approach for mobile robots based on improved sparrow search algorithm. *Adv. Manuf.* **2022**, *10*, 114–130. [CrossRef]
30. Liu, Q.; Zhang, Y.; Li, M.; Zhang, Z.; Cao, N.; Shang, J. Multi-UAV path planning based on fusion of sparrow search algorithm and improved bioinspired neural network. *IEEE Access* **2021**, *9*, 124670–124681. [CrossRef]
31. Low, E.S.; Ong, P.; Low, C.Y.; Omar, R. Modified Q-learning with distance metric and virtual target on path planning of mobile robot. *Expert Syst. Appl.* **2022**, *199*, 117191. [CrossRef]
32. Bonny, T.; Kashkash, M. Highly optimized Q-learning-based bees approach for mobile robot path planning in static and dynamic environments. *J. Field. Robot.* **2022**, *39*, 317–334. [CrossRef]
33. Oh, E.; Lee, H. Effective route generation framework using quantum mechanism-based multi-directional and parallel ant colony optimization. *Comput. Ind. Eng.* **2022**, *169*, 108308. [CrossRef]
34. Tsai, C.-C.; Huang, H.-C.; Chan, C.-K. Parallel elite genetic algorithm and its application to global path planning for autonomous robot navigation. *IEEE Trans. Ind. Electron.* **2011**, *58*, 4813–4821. [CrossRef]
35. Han, B.; Ming, Z.; Zhao, Y.; Wen, T.; Xie, M. Comprehensive risk assessment of transmission lines affected by multi-meteorological disasters based on fuzzy analytic hierarchy process. *Int. J. Electr. Power Energy Syst.* **2021**, *133*, 107190. [CrossRef]
36. Park, Y.; Lee, S.-W.; Lee, J. Comparison of fuzzy AHP and AHP in multicriteria inventory classification while planning green infrastructure for resilient stream ecosystems. *Sustainability* **2020**, *12*, 9035. [CrossRef]

37. Chen, X.; Liu, S.; Liu, R.W.; Wu, H.; Han, B.; Zhao, J. Quantifying Arctic oil spilling event risk by integrating an analytic network process and a fuzzy comprehensive evaluation model. *Ocean Coast. Manag.* **2022**, *228*, 106326. [CrossRef]
38. WU GX, W.L.; ZHENG, J. Dynamic route planning method for intelligent ships considering complex meteorological changes. *J. Shanghai Marit. Univ.* **2021**, *42*, 1–6.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.