THE JOURNAL OF NAVIGATION (2019), **72**, 430–446. © The Royal Institute of Navigation 2018
doi:10.1017/S0373463318000814

Automatic Ship Routing with High Reliability and Efficiency between Two Arbitrary Points at Sea

Shuaidong Jia, Zeyuan Dai and Lihua Zhang

(*Department of Military Oceanography and Hydrography & Cartography, Dalian Naval Academy, Dalian, China; Key Laboratory of Hydrographic Surveying and Mapping of PLA, Dalian Navy Academy, Dalian, China*)
(E-mail: carl-day@foxmail.com)

Due to the limitations of the existing methods (for example, the route binary tree method) that can only automatically generate routes based on a single chart, a method for automatically generating the shortest distance route based on an obstacle spatial database is proposed. Using this proposed method, the route between two arbitrary points at sea can be automatically generated. First, the differences in accuracy and updating time of charts are quantitatively analysed. Next, the mechanism for updating obstacles is designed, an obstacle spatial database is constructed, and the obstacle data extracted from multiple charts are fused. Finally, considering the effect of efficiency on the amount of obstacle data, a route window and an improved R-tree index are designed for quickly extracting and querying the obstacle database. The experimental results demonstrate that compared with existing methods, the proposed method can generate the shortest distance between two arbitrary points at sea and eliminates the limitation of the area of the chart. In addition, with data from multiple charts, the route generated by the proposed method is more reliable than that of the existing methods, and it is more efficient.

KEYWORDS

1. Electronic Navigational Chart. 2. Route Planning. 3. Shortest Path. 4. Charting.

Submitted: 18 March 2018. Accepted: 20 September 2018. First published online: 28 November 2018.

1. INTRODUCTION. A nautical chart, the fundamental carrier of marine geographical information, is vital for the navigational safety of ships (Reed and Schmidt, 2016; Yu et al., 2017; Ladue and Tetreault, 2017; Zhang, 2011). For the complex and changeable ocean environment, an important application of the chart data is the effective design of an optimal route (Chang et al., 2004; Zhang et al., 2011; Lin et al., 2013). The optimal route means the route generated is the best under the existing conditions for ensuring the ship's safe navigation to a specific location (Zhang et al., 2008; Panigrahi et al., 2012; Montes, 2005).

To ensure a ship navigates safely and reaches its destination quickly, many scholars have researched methods for discovering the shortest route. Some typical algorithms and improvements that are used to solve this problem include the Dijkstra algorithm, the Puzzle algorithm, the A* algorithm, the Genetic algorithm and the Theta algorithm (Dijkstra, 1959; Ye and Yu, 2003; Wang et al., 2005; Szlapczynski, 2006; Li et al., 2007; Ying et al., 2007; Nash et al., 2014; Kim et al., 2014; Lee and Kim, 2014; Wijayaningrum and Mahmudy, 2016). However, the routes generated by these algorithms are optimal only for the route network constructed on the basis of the chart, not for reality. Recently, considering the unique advantages of the vector Electronic Navigational Chart (ENC) representation, some scholars have explored building the shortest route based on this medium. Zhang et al. (2011) proposed a method for building the shortest route by constructing rules of obstacle avoidance based on vector ENCs. Wang et al. (2010) proposed a method for designing a route based on a route binary tree to solve the "greediness" in Zhang's (2011) algorithm. Cao et al. (2011) further improved the route binary tree algorithm to improve the efficiency and quality of the route. Then, Wang et al. (2016a; 2016b) considered the influence of the ship's beam and turning characteristics and proposed a solution based on the route binary tree algorithm. Generally, the route binary tree algorithm is capable of designing a route on a vector ENC. However, the current algorithms are limited to a single chart when designing the route. Similar to the limitation of the scope of routes covered by a nautical chart, it is generally difficult to cover the entire sea area that a ship navigates using only one nautical chart. Therefore, a small scale chart should be utilised for generating a route automatically. Obviously, the smaller the chart scale, the lower the chart accuracy, and accordingly, the less reliable and safe the route.

To solve the above-mentioned problems, a method for automatically building the shortest distance route based on an obstacle spatial database is proposed in this paper. The mechanism for fusing and updating the obstacle data from different charts (in this paper, the chart used is the ENC) is designed, and an obstacle spatial database for navigation is established. Then, based on the algorithm of the route binary tree, a route window and improved R-tree index algorithm are proposed for quickly searching and extracting obstacles, thereby increasing the efficiency of building the route.

2. THE BASIC THEORY OF THE EXISTING METHOD.

2.1. *Basic idea.* In recent research on automatic route generation, the algorithm of the route binary tree based on a single chart is one of the most common approaches and its basic idea is as follows: First, a representation of the obstacles that threaten the safety of ship navigation is constructed by analysing and modelling chart data. Then, the mechanism of automatically keeping away from the obstacles is designed and thus, a feasible route is generated.

2.2. *Obstacle representation based on a single chart.* To automatically generate safe and reliable routes, the concept of obstacles is proposed and used to describe the sea area where the ship cannot navigate (Zhang, 2011; Chang et al., 2004; Szlapczynski, 2006). Based on the existing studies, the obstacles mainly include shallow depth obstacles (for example, land area, island area, rocks) and man-made obstacles (for example, prohibited area, shipwreck area) (Zhang et al., 2011). These obstacles can be extracted from the chart. The specific method is as follows.

First, the shallow areas are extracted. Based on the sounding points from the chart, a depth model can be constructed with a Triangulated Irregular Network (TIN) (Zhang et al., 2008). Considering the ship's draft, the safe depth contours are traced from the depth

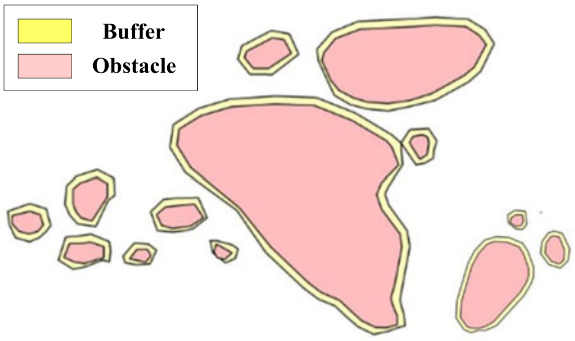


Figure 1. Representation of obstacles in the chart area.

model, and thus the shallow depth areas are acquired (Zhang, 2011; Zhang et al., 2008; Peters et al., 2014; Guilbert, 2016).

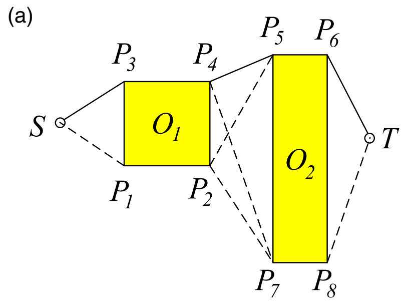
Second, the man-made areas are extracted. According to the chart area data attributes, man-made obstacles can be easily extracted from the chart.

The ocean environment is complex and changeable, and some factors (for example, the ship's circumflex radius and positioning accuracy) must be considered when designing routes (Pietrzykowski and Uriasz, 2009; Jones and Rowe, 2016). Therefore, obstacle areas should be extended with a certain buffer. As some obstacle areas may intersect after the buffer is added, the obstacle areas need to be assessed and simplified to ensure the validity of the topological relationships of the obstacle areas. As shown in Figure 1, the obstacle areas are represented in one chart area.

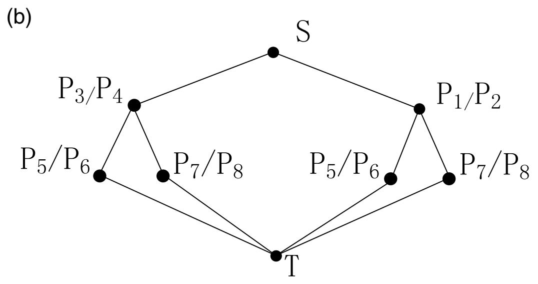
2.3. *Automatic route generation based on the route binary tree.* After extracting obstacle areas, the next step is to generate a route. The automatic generation of the shortest distance route means finding the shortest path from the starting point to the destination, for which the classic method is the route binary tree algorithm (Wang et al., 2005; Cao et al., 2011; Wang et al., 2016a; 2016b).

As shown in Figure 2(a), the starting point is S, the destination point is T, and the obstacle areas are O_1 and O_2. The basic principle of the route binary tree algorithm is as follows:

(1) Take the starting point S as the current test point and find the nearest obstacle area O_1 from which the vessel at the current test point needs to keep away.
(2) Determine the left child node P_1 and the right child node P_3 and establish the route binary tree.
(3) Regard the child nodes P_1 and P_3 as the current test points, and repeat steps (1) and (2) until there are no obstacle areas from the test points to the terminal point T.
(4) Build a route binary tree as shown in Figure 2(b), and then calculate the distance between the nodes and store it in the data structure of the route binary tree.
(5) Search the data structure of the route binary tree from top to bottom and sum up the distance of all optional routes and select the shortest one. For this example, the shortest route is $SP_3P_4P_5P_6T$.



Process of Generating the Route Binary Tree



Framework of the Route Binary Tree

Figure 2. Route Generation Based on the Route Binary Tree.

3. BUILDING THE SHORTEST ROUTES BASED ON THE OBSTACLE SPATIAL DATABASE.

3.1. *Basic idea.* The existing binary tree algorithm can generate the shortest route between any two points on a single chart. However, one chart alone is generally not enough for a ship's navigation. While two (or more) charts are utilised, the existing algorithms cannot be adopted to generate routes. Therefore, a new method is proposed in this paper, and its basic idea is as follows:

(1) *Fuse the obstacle data extracted from different charts*. Obstacle data extracted from different charts are compared and analysed quantitatively, and the model fusing all the obstacle data is constructed. Thus, the obstacles from different charts can be represented together with a high accuracy.

(2) *Building and updating the obstacle spatial database*. A spatial database is built for storing obstacle data fused from the different charts. In addition, considering the influence of chart correction and reproduction, a mechanism is designed for sorting and updating the obstacle data.

(3) *Optimising the efficiency of the algorithm*. With the obstacle data fused from multiple charts, the amount of data may become increasingly large. Considering the influence of the amount of data on the algorithm, a route window and an improved R-tree index are proposed for quickly searching and extracting the obstacles.

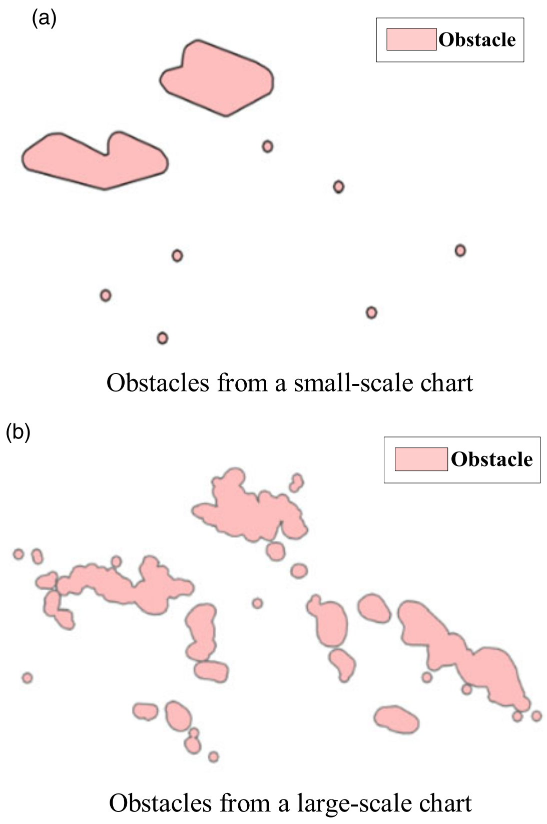

(a)
Obstacle
Obstacles from a small-scale chart
(b)
Obstacle
Obstacles from a large-scale chart

Figure 3. Representation of obstacles extracted from different charts.

3.2. *Fusing the obstacle data extracted from the different charts.* A small-scale chart cannot accurately preserve all the data from a large-scale chart (Yan et al., 2016). Therefore, a cartographic generalisation should be performed to simplify the data from the large-scale chart to be compatible with that from a small-scale chart, and hence the accuracy is decreased.

As shown in Figure 3, the obstacles extracted from two charts with different scales represent the same sea area. The obstacles extracted from the chart with a small scale are shown in Figure 3(a), and the obstacles from the chart with a large scale are shown in Figure 3(b). It can be concluded that, the larger the chart scale, the higher the chart accuracy, but the smaller the sea area covered; on the other hand, the smaller the chart scale, the larger the sea area covered, but the lower the chart accuracy. In Figure 3, it is shown that some navigable areas such as the narrow channel and waterway shown in Figure 3(b) are represented as non-navigable areas in Figure 3(a), and even some non-navigable areas in Figure 3(b) are represented as navigable areas in Figure 3(a), so that the navigational safety of a ship might be threatened. If the distance between the start point and the end point is so far that the chart with the large scale cannot overlay both points, the existing method for generating a ship route only utilises the chart with the small scale, which can overlay both the start point and the end point, thereby decreasing the reliability of the generated route.

For generating a route with high reliability, a model for fusing the obstacle data extracted from different charts is constructed in this paper. The specific method is as follows:

(1) For quantitatively comparing the accuracy of two (or more) charts, an index of accuracy is defined. Usually, the chart accuracy is positively related to the chart scale. In other words, the larger the chart scale, the higher the chart accuracy, which can be expressed as follows:

$$\begin{cases} S_{(A)} = F_{(\varepsilon_1,\mu_1)} = \{a_0, a_1, a_2 \ldots a_m\} \\ S_{(B)} = F_{(\varepsilon_2,\mu_2)} = \{b_0, b_1, b_2 \ldots b_n\} \end{cases} \tag{1}$$

where $S_{(A)}$ and $S_{(B)}$ represent the dataset of the obstacles from the charts A and B, respectively. $F(\varepsilon, \mu)$ represents the obstacle function, and its parameters ε and μ represent the sea area covered and the chart accuracy, respectively. In addition, a_i and b_j represent the i-th and j-th obstacles extracted from charts A and B, respectively.

(2) The intersected area of two (or more) charts means the sea area is represented by different charts. In this type of area, obstacles extracted from different charts are usually different. Therefore, the obstacles need to be fused as follows:

$$\begin{cases} \varepsilon_0 = \varepsilon_1 \cap \varepsilon_2 \\ S_{(A_0)} = F_{(\varepsilon_0,\mu_1)} = S_{(A)} \cap \varepsilon_0 = \{a_i, a_{i+1}, a_{i+2} \ldots a_{i+k}\} \\ S_{(B_0)} = F_{(\varepsilon_0,\mu_2)} = S_{(B)} \cap \varepsilon_0 = \left\{a_j, a_{j+1}, a_{j+2} \ldots a_{j+h}\right\} \end{cases} \tag{2}$$

where $S_{(A_0)}$ and $S_{(B_0)}$ represent the set of obstacles from the intersecting areas from charts A and B, respectively. ε_0 represents the intersecting areas from charts A and B. a_x $(x \in Z, i \leq x \leq i+k)$ and b_y $(y \in Z, j \leq y \leq j+h)$ represent the obstacles a_x and b_y respectively, located inside the area ε_0.

(3) On the other hand, for the area not intersecting with chart A or B (covered by only one chart), the obstacles cannot be ignored and are represented as follows:

$$\begin{cases} S_{(A')} = F_{(\varepsilon_1,\mu_1)} - F_{(\varepsilon_0,\mu_1)} = \{a_0, a_1, a_2 \ldots a_{i-2}, a_{i-1} \ldots a_{i+k+1} \ldots a_m\} \\ S_{(B')} = F_{(\varepsilon_2,\mu_2)} - F_{(\varepsilon_0,\mu_2)} = \left\{b_0, b_1, b_2 \ldots b_{j-2}, b_{j-1} \ldots b_{j+h+1} \ldots b_n\right\} \end{cases} \tag{3}$$

where $S_{(A')}$ and $S_{(B')}$ represent the obstacles extracted from the charts A and B, respectively.

(4) Considering Equations (2) and (3), the obstacles fused from charts A and B can be represented as follows:

$$S_{(C)} = \begin{cases} S_{(A')} + S_{(B')} + S_{(A_0)} = F_{(\varepsilon_1,\mu_1)} + F_{(\varepsilon_2,\mu_2)} - F_{(\varepsilon_0,\mu_2)} & (\varepsilon_0 \neq 0, \mu_1 \geq \mu_2) \\ S_{(A')} + S_{(B')} + S_{(B_0)} = F_{(\varepsilon_1,\mu_1)} - F_{(\varepsilon_0,\mu_1)} + F_{(\varepsilon_2,\mu_2)} & (\varepsilon_0 \neq 0, \mu_2 > \mu_1) \\ S_{(A')} + S_{(B')} = F_{(\varepsilon_1,\mu_1)} + F_{(\varepsilon_2,\mu_2)} & (\varepsilon_0 = 0) \end{cases} \tag{4}$$

where $S_{(C)}$ represents the obstacle set fused from charts A and B.

By utilising this series of steps for more charts, obstacles extracted from different charts can be fused.

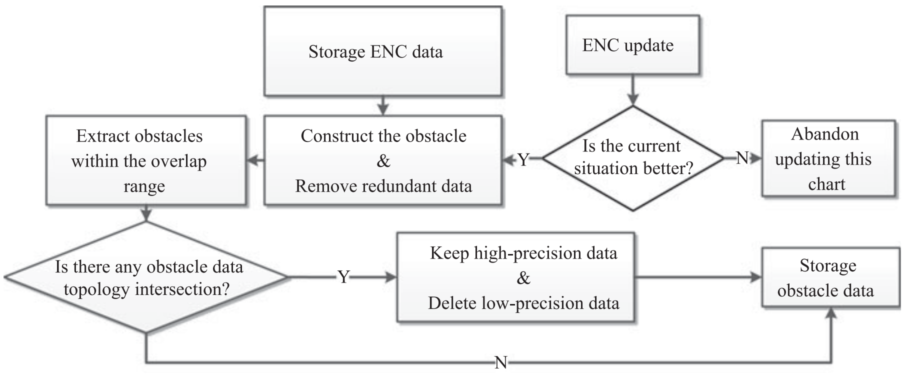

Storage ENC data
ENC update
Extract obstacles within the overlap range
Construct the obstacle & Remove redundant data
Y
Is the current situation better?
N
Abandon updating this chart
Is there any obstacle data topology intersection?
Y
Keep high-precision data & Delete low-precision data
Storage obstacle data
N

Figure 4. The flow diagram of obstacle data fusion and updating.

3.3. *Construction and update of obstacle spatial database.* After the obstacle data extracted from different charts has been fused, the amount of data will increase. Therefore, a method for managing the obstacle data is needed, and in this paper, an obstacle spatial database is constructed. Considering the charts that need to be updated, the approach to updating the obstacle spatial database is designed.

For quantitatively analysing the updating time of the chart, Equation (1) needs to be further expanded as follows:

$$S = F(\varepsilon, \mu, t) \tag{5}$$

where t represents the index of the chart updating time, which is positively related to the time of chart publication. The more recently the chart was published, the higher the index of the chart updating time.

On the basis of the work in Section 2.2, considering how recent the chart is, the obstacle spatial database needs to be updated regularly. As shown in Figure 4, the basic flow of updating is as follows:

(1) Extract the obstacles from single charts and fuse them.
(2) If a chart is updated, delete the raw obstacles from the obstacle spatial database and replace them with the new obstacles.
(3) If a chart is newly amended, extract the obstacles from the chart and fuse them with the existing obstacles in the spatial database.

3.4. *Efficiency optimisation.* In the existing route binary tree algorithm, many topological calculations are conducted (Wang et al., 2005; Cao et al., 2011; Wang et al., 2016a; 2016b). For example, in the existing algorithm, all the obstacles need to be cycled many times and calculated one by one. With the fusing and updating of the obstacle data from many charts, the more the data increases, the lower the efficiency of the algorithm becomes. To solve this problem, two ways for enhancing the efficiency of the algorithm are proposed in this paper. The obstacle areas can be extracted quickly by route window and queried efficiently based on the improved R-tree.

3.4.1. *Obstacle areas quickly extracted by route window.* As described in Sections 2.2 and 2.3, all the obstacle data for generating high-precision routes can be fused and updated from multiple charts. However, the size of the area is large, the topological relationship is complex, and the area of indication is wide. If the area can be limited to a certain degree

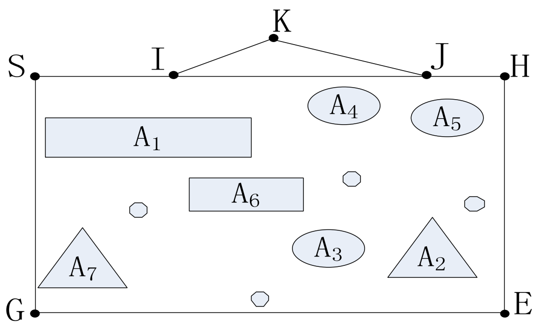


Figure 5. Route window and obstacles inside.

for the algorithm of the route binary tree, the data size will be reduced for the calculation, so that the efficiency of the route generation is enhanced.

According to the spatial theory of the rolling window, the route window is defined as follows. First, the connection between the starting point and the ending point of the route is established. Next, the line segment from the starting point to the ending point is regarded as the diagonal of a certain rectangle in space. Finally, the rectangle defined by this diagonal is called the route window. If there is a route that intersects with the route window topology, the route window needs to be expanded to include that route. As shown in Figure 5, if the starting point is S and the ending point is E, then the route window is the rectangle $SHEG$, and the obstacles intersecting with the route window are $A_1, A_2, \ldots, A_7$.

The route window has the property that the route generated by the binary tree algorithm must be located within it.

The above conclusion can be drawn by this method:

As shown in Figure 5, set the point S as the starting point and point E as the ending point. Then, rectangle $SHGE$ is a route window defined by diagonal SE. If the route contains a point K out of the route window, which intersects with the window by points I, J, it is easy to see that:

$$IK + KJ > IJ \tag{6}$$

Additionally, the route IJ is navigable, which means the shortest route does not pass through the point K.

In complex sea areas, route windows can reduce the number of obstacle courses incorporated into the calculation, thus the efficiency of the algorithm is improved.

3.4.2. *Efficient obstacle query based on the improved R-tree.* The theory of the R-tree index is usually utilised for quickly querying geospatial data. The conventional R-tree consists of a root node, an intermediate node and a leaf node. The intermediate node represents a rectangle in the spatial data, which is the Minimum Bounding Rectangle (MBR) of all its "children," not a geographical object in the real world. Leaf nodes are geographical entities with a practical significance. As the R-tree index is actually an extension of the B-tree index in a spatial dimension and replaces the complex space object with an MBR, it not only has the unique dynamic balance of a B-tree but also becomes flexible and efficient for a large amount of spatial data.

However, the spatial query efficiency based on the R-tree index is positively correlated with the overlap area of the MBR constructed, and the overlap of the MBR becomes larger

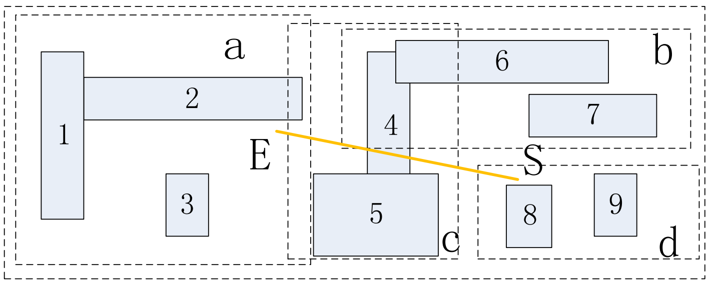


Figure 6. Spatial query based on a traditional R-tree.

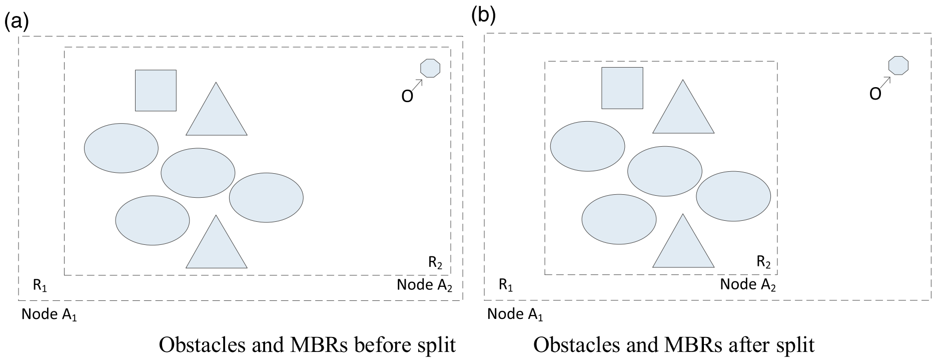

(a)
(b)
o
R_2
R_1
Node A_2
Node A_1
Obstacles and MBRs before split
Obstacles and MBRs after split

Figure 7. The model of a node split.

after the data is updated and inserted. During route generation, the spatial query operation is usually performed linearly. When the spatial query is performed, the larger the overlapping areas of the MBRs, the more child nodes are returned and the greater the number of the sub-trees that need to be searched. As shown in Figure 6, the MBRs intersecting the line SE during the spatial query include a, b, c and d. In accordance with the query algorithm, all the child nodes in the four sub-trees need to be traversed. However, none of a, b or d's sub-trees contain the leaf node where the real object eventually needs to be located, which greatly affects the time overhead of the spatial query. To solve the overlap problem of MBRs, the conventional R-tree is improved through splitting child nodes to improve the density of objects in the MBR.

In the process of node splitting, the main idea is to store the divisible boundary objects in the child nodes in the regions of their parent nodes. Figure 7 shows the model before and after node splitting. For detail, node A_1 and its child node A_2 are used as an example as shown in Figure 7. In this figure, when object o needs to be queried, A_1 and A_2 both need to be queried.

The method we used to reach the research objective is as follows. First, for node A_2, find the geographic objects (sometimes it may be a collection of objects with its own MBR, similar to a small child node) that are far away from other child nodes. In this example, it is object o. Next, split the objects from the original nodes and reduce the area of the MBR. As shown in the figure, we get a smaller MBR for node A_2, R_2. Finally, store the object in node A_1, and store the new MBR in A_2.

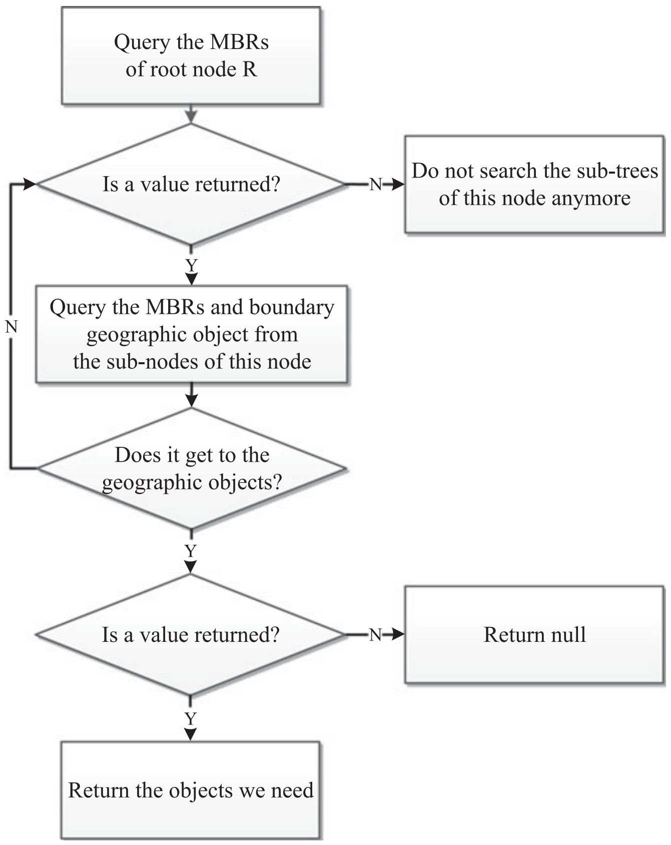

Query the MBRs of root node R
Is a value returned?
N
Do not search the sub-trees of this node anymore
Y
Query the MBRs and boundary geographic object from the sub-nodes of this node
N
Does it get to the geographic objects?
Y
Is a value returned?
N
Return null
Y
Return the objects we need

Figure 8. The flowchart of obstacle querying based on an improved R-tree.

After node splitting, when querying object o, we only query node A_1. This reduces the area of the MBR and increases the spatial query efficiency.

It can be seen that after splitting A_2, the MBR of the three new nodes is significantly reduced compared to the original area. Of course, in practice, it is impossible to divide the nodes blindly, so there is no difference between the formed R tree and the traversal data. According to the characteristics of chart data, it is stipulated that the new features stored after splitting cannot be less than $M/4$, where M is the maximum space of the nodes.

Spatial data indexed by the improved R-tree can be divided into a series of steps as shown in the flowchart above when the topology operation is performed as follows (Figure 8):

(1) Start from the root node R and perform the corresponding topology operation. If it returns true, proceed to (2); otherwise, go to (5).
(2) Traverse the MBRs and boundary geographic objects represented by each child node under the root node for the corresponding topology operation. If it returns true, continue to step (3); otherwise, go to (5).
(3) Traverse the MBRs and boundary geographic objects represented by all the child nodes at the next level of the child node for the corresponding topology operation (3)

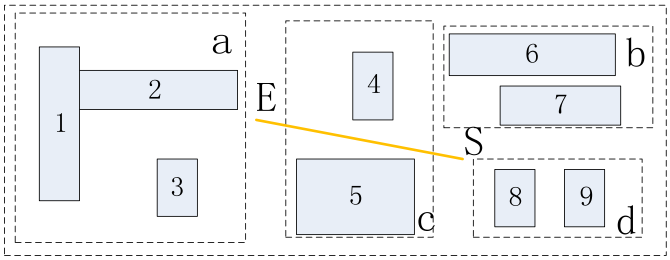


Figure 9. Spatial query based on an improved R-tree.

until the leaf node at the bottom of the R tree (the real obstacle data in this paper) is reached, proceed to (4); if it returns a null value, go to (5)

(4) Carry out the corresponding topology operation on all the real geographical entities in the obtained sub-tree to get the data that meets the requirements and store it; if it cannot be found, an empty set shall be returned.

(5) If the corresponding topology operation returns a null value, it is impossible for all the real geographic entities corresponding to its child nodes and the child nodes to intersect, and the search for the sub-tree is no longer performed.

For the area shown in Figure 6, for example, the spatial query diagram after the spatial splitting is shown in Figure 9. The line ES only passes through the rectangle c, which greatly reduces the number of sub-trees searched and improves the efficiency of the query process.

4. EXPERIMENT AND ANALYSIS.

4.1. *Comparison and analysis of obstacles.* To compare and analyse the difference in the obstacle representation between the existing algorithm and the algorithm proposed in this paper, the two methods above are used for constructing the obstacles inside the experimental area. The charts relevant to the experimental area are shown in Table 1. As the existing algorithm cannot find one suitable chart with a large scale covering the experimental area, only the data from the chart No. C1100104 (scale 1:2,300,000) can be used for constructing the obstacle database. On the other hand, the proposed algorithm constructs the obstacle database for each chart and fuses the data from different charts, and then, the obstacles relevant to the experimental area are extracted. The experimental results are shown in Figures 10 and 11.

As shown in Figures 10 and 11, the small-scale chart used in the binary tree for generating the navigation route has a large amount of cartographic synthesis of all the elements in the chart due to the limitation of the load carrying capacity of the map, and it is inexact. Take the sea area shown in the dotted line box A as an example. Some actually existing obstacle areas on the small-scale chart are not shown in the figure, which would threaten the navigational safety of the ship. In the dotted line box B on the left of the sea area, there is a situation in which the route is blocked in the navigation area after the cartographic synthesis is completed, and the generated route is no longer the actual shortest route.

Compared with the existing method, the proposed method is more detailed in its construction and expression of the obstacles, as shown in Figure 11. This is because the proposed algorithm not only accurately represents the actual obstruction area (as shown

Table 1. Charts relevant to the experimental area

Chart Number	Chart Name	Scale
C1100104	South China Sea	1:2,300,000
C1315700	Hailing Channel to Baohu Cape	1:250,000
C1415710	Dafangji to Weizhou Island	1:120,000

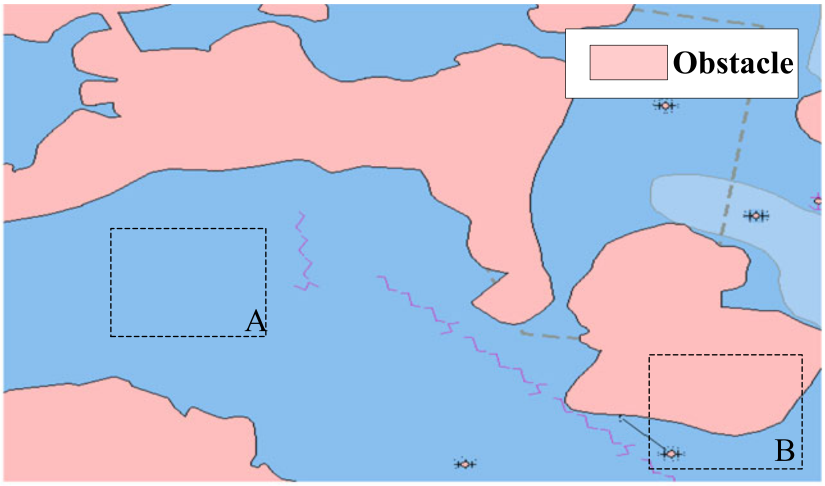


Figure 10. Obstacles extracted from a single chart.

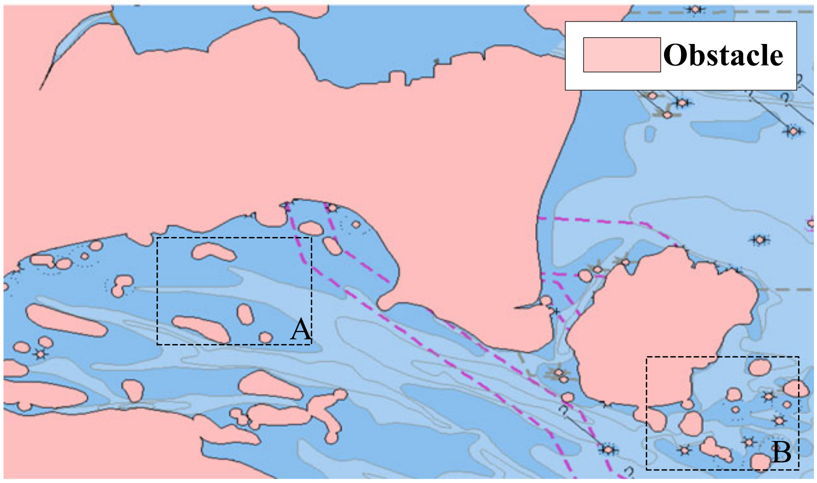
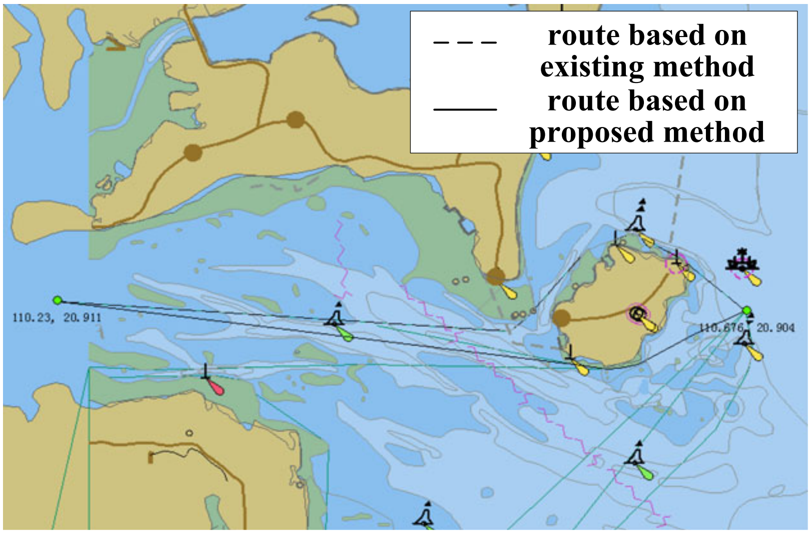


Figure 11. Obstacles fused with multiple charts.

in the area A in Figure 11) by data fusion of the sea areas established from different scale charts but also shows the navigable waterway in usable detail (shown as area B in Figure 11).

Table 2. Relevant charts for generating routes.

No.	Chart Number	Scale	Does route intersect with the chart area
1	C1100104	1:2,300,000	No
	C1315700	1:250,000	Yes
	C1415710	1:120,000	Yes
2	C1100103	1:2,300,000	No
	C1210013	1:1,000,000	No
	C1413370	1:150,000	Yes
	C1413510	1:150,000	Yes


route based on existing method
route based on proposed method
110.23, 20.911
110.676, 20.904
Figure 12. Comparison of routes generated by the different methods (Experiment No. 1).

4.2. *Comparison and analysis of routes generated by different algorithms.* Next, the existing route binary tree method and the method proposed in this paper are further compared and analysed for route generation. Two sets of electronic chart data are selected as shown in Table 2 and using the above two methods to plan the route, the test results are shown in Figures 12 and 13. For the ease of comparison, a partial area C of Figure 13 is enlarged and displayed in Figure 14.

From the test results shown in Figures 12 and 13, when compared with the existing route binary tree algorithm, the distance of the route generated by the proposed method is obviously shorter than that of the existing algorithm. The proposed method is able to plan the shortest distance route based on the presentation of a finer obstacle course to make full use of the navigable space.

As further shown in Figure 14, which is an enlarged view of area C in Figure 13, the route, which is generated by the existing binary tree algorithm using the small-scale chart, goes across the obstacle area in the large-scale chart. This has the potential to hide dangers to navigation safety.

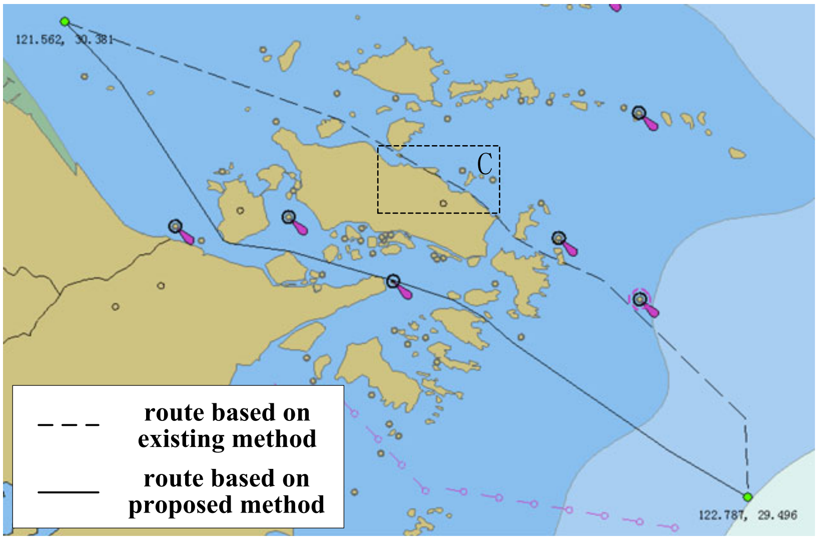

121.562, 30.381
C
122.787, 29.496
route based on existing method
route based on proposed method

Figure 13. Comparison of routes generated by the different methods (Experiment No. 2).

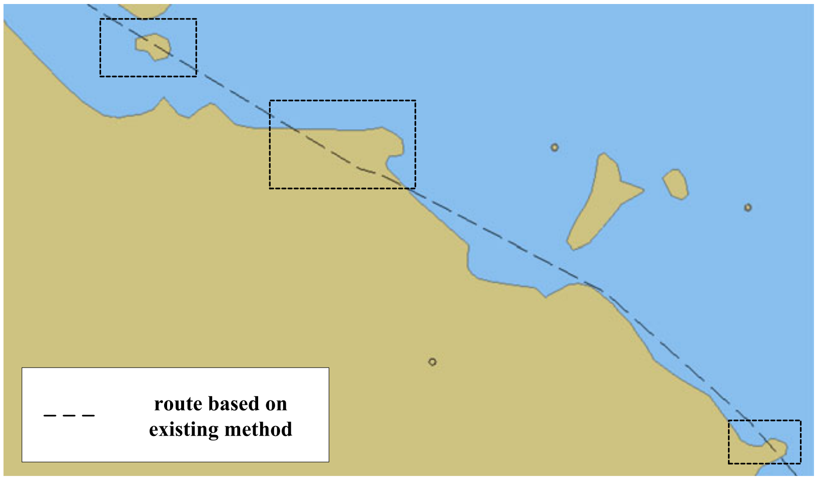


Figure 14. Route in area C.

Generally, the results of the experiment show that compared with the existing route binary tree algorithm, the shortest distance route generated by the method proposed in this paper has been greatly improved both in distance and in safety. The feasibility and reliability of this route can be concluded as shown in Table 3.

4.3. *Comparison and analysis of efficiencies of different algorithms.* The difference in the efficiency between the existing route binary tree algorithm and the proposed method is further compared and analysed. The two groups of the shortest distance routes are selected as shown in Figure 11, the basic parameters of the PC used for the test are shown in Table 4, and the efficiency of the test results are shown in Table 5.

Table 3. Comparison of feasibility and reliability.

No.	Methods	Route Navigability	Reliability	Route Distance (nm)
1	Existing Method	Un-navigable	Low	28·72
	Proposed Method	Navigable	High	25·71
2	Existing Method	Un-navigable	Low	88·56
	Proposed Method	Navigable	High	86·62

Table 4. Hardware configuration for the test.

Equipment	Processor	RAM	Hard Disk	Graphics/Memory	Operating system
PC	Intel core i5-3230 M 2.60 GHz	8GB	1TB	AMD 7400 M/1GB	Windows7 x64

Table 5. Comparison of efficiency.

No	Obstacle Number	Methods	Running Time (s)
1	98	Existing Method	30·15
		Proposed Method	5·62
2	153	Existing Method	62·30
		Proposed Method	8·18

As seen from Table 5, when compared with the existing route binary tree algorithm, the method proposed in this paper significantly shortens the route generation time. This is due to the effective screening of the obstacle course data in the route window, the improvement of the R-tree index and the efficient spatial query of the obstacle course data. Hence, the efficiency is improved.

5. CONCLUSIONS.

The analysis, calculation and comparison lead to the following conclusions:

(1) Compared with the existing algorithm, which only represents obstacles extracted from a single chart, the method proposed in this paper can fuse obstacles from different charts and represent them by analysing the differences in the accuracy and updating time of different charts. Therefore, the obstacle databases constructed with the proposed method are more accurate than those of the earlier method.

(2) As the accuracy of the obstacle database constructed with the proposed method is better than the existing method, the route generated by the proposed method is more feasible and reliable than that of the existing method.

(3) Compared with the existing method for generating a route automatically, the route window and the improved R-tree approach proposed in this paper have greatly improved the efficiency of automatic route generation under a large amount of obstacle data, and the route generation time is greatly shortened.

It is worth mentioning that the method proposed in this paper can use all vector formatted charts, rather than merely Electronic Navigational Chart (ENC) data. Additionally, the data fusion of the obstacles only considers the two basic elements of accuracy and the current situation. A data fusion with obstacles under more complex situations (such as charts issued by different countries) remains to be further analysed. Moreover, if the route is much longer and the data size of the obstacles is much greater, generating a route with a high reliability and a high efficiency will need to be further studied.

ACKNOWLEDGMENTS

This paper is supported by National Natural Science Foundation of China (41471380, 41601498 and 41774014). We would like to thank the anonymous reviewers for their valuable suggestions.

REFERENCES

Cao, H.B., Zhang, L.H., Jia, S.D. and Zhang, L. (2011). An Improved Method for Automatically Building Shortest Route Based on Electronic Chart, *Geomatic and Information Science of Wuhan University*, **36**(9), 1107–1110.

Chang, K., Jan, G.E. and Parberry, I. (2004). A Method for Searching Optimal Routes with Collision Avoidance on Raster Charts. *The Journal of Navigation*, **56**(3), 371–384.

Dijkstra E. (1959). A note on two problems in connection with graphs. *Numerische Mathematic*, **1**, 269–271.

Guilbert, E. (2016). Feature-Driven Generalization of Isobaths on Nautical Charts, A Multi-Agent System Approach. *Transactions in GIS*, **20**(1), 126–143.

Jones, M. and Rowe, R. (2016). Collision avoidance system. *US Patent*, US 9,505,404 B2.

Kim, H., Kim, D., Shin, J.U., Kim, H. and Myung, H. (2014). Angular rate-constrained path planning algorithm for unmanned surface vehicles. *Ocean Engineering*, **84**(4), 37–44.

Ladue, D. and Tetreault, B. (2017). Improving navigation safety with better charts, Prototyping AIS for Ship Reporting of ENC Discrepancies. *Sea Technology*, **58**(3), 20–24.

Lee, Y.H. and Kim, S.W. (2014). A Hybrid Search Method of A* and Dijkstra Algorithms to Find Minimal Path Lengths for Navigation Route Planning. *Journal of the Institute of Electronics & Information Engineers*, **51**(10), 109–117.

Li, Y.H., Pan, M.Y. and Wu, X. (2007). Automatic creating algorithm of route based on dynamic grid model. *Journal of Traffic and Transportation Engineering*, **7**(3), 34–39.

Lin, Y.H., Fang, M.C. and Yeung, R.W. (2013). The optimization of ship weather-routing algorithm based on the composite influence of multi-dynamic elements. *Applied Ocean Research*, **43**(43), 184–194.

Montes, A.A. (2005). Network Shortest Path Application for Optimum Track Ship Routing. *California, Naval Postgraduate School*.

Nash, A., Daniel, K., Koenig, S. and Felner, A. (2014). Theta*, Any-Angle Path Planning on Grids. *Journal of Artificial Intelligence Research*, **39**(1), 533–579.

Panigrahi, J.K., Padhy, C.P., Sen, D., Swain, J. and Larsen, O. (2012). Optimal ship tracking on a navigation route between two ports, a hydrodynamics approach. *Journal of Marine Science & Technology*, **17**(1), 59–67.

Peters, R., Ledoux, H., Meijers, M. (2014). A Voronoi-Based Approach to Generating Depth-Contours for Hydrographic Charts. *Marine Geodesy*, **37**(2), 145–166.

Pietrzykowski, Z. and Uriasz, J. (2009). The Ship Domain - A Criterion of Navigational Safety Assessment in an Open Sea Area. *The Journal of Navigation*, **62**(1), 93–108.

Reed, S. and Schmidt, V.E. (2016). Providing Nautical Chart awareness to autonomous surface vessel operations. *OCEANS 2016 MTS/IEEE Monterey*. Monterey, CA. 1–8.

Szlapczynski, R. (2006). A New Method of Ship Routing on Raster Grids, with Turn Penalties and Collision Avoidance. *The Journal of Navigation*, **59**(1), 27.

Wang, D.C., Chen, L.M. and Zhang, X.F. (2005). Selecting Ship's Optimum Route Using A* Algorithm. *Journal of Qingdao University (Natural Science Edition)*, **18**(4), 10–13.

Wang, T., Zhang, L.H., Peng, R.C., Cao, H.B. and Jiang, L.J. (2016a). A Method for Automatically Generating the Shortest Distance Route Based on Electronic Navigational Chart Considering Channel Width. *Hydrographic Surveying and Charting*, **36**(3), 29–31, 36.

Wang, T., Zhang, L.H., Peng, R.C., Cao, H.B. and Jiang, L.J. (2016b). Automatic generation of the shortest route of an electronic navigational chart considering turning restrictions. *Journal of Harbin Engineering University*, **37**(7), 923–929.

Wang, Z., Li, S.J., Zhang, L.H. and Li, N. (2010). A method for Automatic Routing Based on Route Binary Tree. *Geomatic and Information Science of Wuhan University*, **35**(4), 407–410.

Wijayaningrum, V.N. and Mahmudy, W.F. (2016). Optimization of Ship's Route Scheduling Using Genetic Algorithm. *Indonesian Journal of Electrical Engineering and Computer Science*, **2**(1), 180–186.

Yan, J., Guilbert, E. and Saux, E. (2016). An ontology-driven multi-agent system for nautical chart generalization.*American Cartographer*, **44**(3), 201–215.

Ye, Q. and Yu, Z.W. (2003). The Improved Shortcut Algorithm and It's Application in Selecting Ship's Optimum Route. *Navigation of China*, **2**, 15–17.

Ying, S., Shi, C. and Yang, S. (2007). Ship Route Designing for Collision Avoidance Based on Bayesian Genetic Algorithm. *IEEE International Conference on Control and Automation*. Guangzhou, CHINA. 1807–1811.

Yu, Y., Zhu, H., Yang, L. and Wang, C.X. (2017). Spatial Indexing for Effective Visualization of Vector-Based Electronic Nautical Chart. *International Conference on Industrial Informatics - Computing Technology, Intelligent Technology, Industrial Information Integration*. Wuhan, CHINA. 323–326.

Zhang, L.H. (2011). *The Methods for Generating The Optimal Route Based on Electronic Navigation Chart*. Science Press.

Zhang, L.H., Zhang, L., Peng, R.C., Li, G.X. and Zou, W. (2011). Determination of the Shortest Time Route Based on the Composite Influence of Multidynamic Elements. *Marine Geodesy*, **34**(2), 108–118.

Zhang, L.H., Zhu, Q., Zhang, A.M., Liu, Y.C. and Han, Y.L. (2008). An intelligent method for the shortest routing. *Acta Geodaetica et Cartographica Sinica*, **1**, 114–120.