Journal of *Marine Science and Engineering*

MDPI

Article

Path Planning Method of Unmanned Surface Vehicles Formation Based on Improved A* Algorithm

Tongtong Sang [1,2,3], Jinchao Xiao [1,2,*], Junfeng Xiong [2], Haoyun Xia [2,4] and Zhongze Wang [1,2,3]

[1] Shenyang Institute of Automation, Shenyang 110016, China
[2] Guangzhou Institute of Industrial Intelligence, Guangzhou 511458, China
[3] University of Chinese Academy of Sciences, Beijing 100049, China
[4] College of Information Engineering, Shenyang University of Chemical Technology, Shenyang 110142, China
* Correspondence: xiaojinchao@sia.cn; Tel.: +86-1812-672-5696

Abstract: Unmanned surface vehicle (USV) formation is a hot topic of current research. Path planning is the core technology for USV formation. This paper focuses on a USV formation path planning problem considering kinetic constraints. Firstly, an improved A* algorithm is proposed to solve the point-to-point path planning of a USV considering kinetic constraints. In this algorithm, the yaw constraint is introduced on top of the position constraint to extend the state space of the USV to three dimensions, and the convergence speed is accelerated by building a heuristic map. The dynamics model of the USV is used to generate the minimum trajectory elements to ensure that the path conforms to the kinetic constraints. Secondly, the mathematical model of USV formation based on the virtual structure method is established, and the path planning scheme of formation navigation and formation reconfiguration is given according to the improved A* algorithm. Finally, we carry out a USV model identification experiment for SL900 USV and simulation experiments based on the model. The experimental results show that the output path of the proposed method is smoother compared with the traditional method. This method can provide a globally safe path with kinetic constraints for USV formation navigation and formation reconstruction.

Keywords: minimal trajectory elements; heuristic value map; formation navigation; formation reconfiguration; model identification experiment

Citation: Sang, T.; Xiao, J.; Xiong, J.; Xia, H.; Wang, Z. Path Planning Method of Unmanned Surface Vehicles Formation Based on Improved A* Algorithm. *J. Mar. Sci. Eng.* **2023**, *11*, 176. https://doi.org/10.3390/jmse11010176

Academic Editor: Michele Viviani

Received: 30 November 2022
Revised: 3 January 2023
Accepted: 8 January 2023
Published: 10 January 2023

1. Introduction

1.1. Background Introduction

The ocean accounts for approximately 70% of the Earth's total area [1]. It is rich in mineral resources and biological resources. It is an important place for human economic, cultural, and scientific research. However, the harsh and changeable marine environment makes us face many difficulties in the process of surveying marine resources and safeguarding maritime rights and interests. The traditional way of using manned vessels often has difficulty ensuring the safety of personnel and has low efficiency. We need to develop auxiliary development platforms for efficient and safe water resources to better control water resources and safeguard maritime rights. As a surface mobile platform with autonomous operation ability, unmanned surface vehicles (USVs) have the advantages of dangerous environmental operation, low cost, and high efficiency, and have attracted extensive attention from countries and economies all over the world [2–5]. USV formation refers to several USVs cooperating to complete one or more tasks in a certain formation. Compared with a single USV, the formation of USVs has stronger robustness, higher operating efficiency, and wider operating range. It can be applied to more complex mission scenarios. The formation of USVs has high practical engineering application value and is a current and future development direction.

USV formation can be divided into three layers: task management layer, formation path planning layer, and task execution layer [6], as shown in Figure 1. The task management layer decomposes and assigns tasks to each USV according to the task requirements and the currently managed USV's state information. The formation task configuration information obtained by each USV includes the task start point, the task endpoint, the formation shape information, the formation reconfiguration information, and the position information of the USV in the formation. The mission management layer passes this formation configuration information to the formation path planning layer. The path planning module plans the motion trajectory of each USV according to the formation configuration information, environmental information, and kinetic model. The formation path planning layer transmits the trajectory of each USV to the task execution layer. The trajectory tracking module calculates the real-time control output according to the trajectory and the pose information of the USV so that the USV can track the trajectory. The control output of the USV is thrust and rudder.

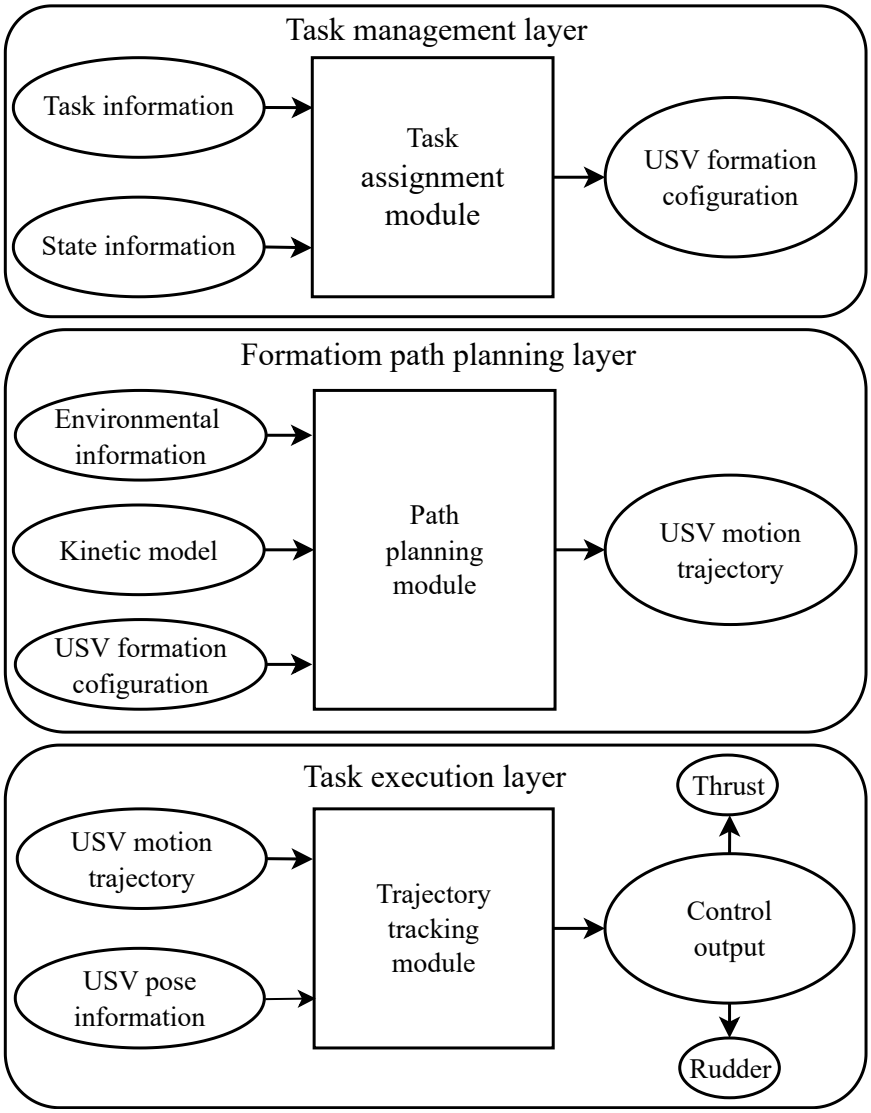

Figure 1. The layered structure of unmanned surface vehicles formation.

From the hierarchical structure of USV formation, we can see that the formation path planning layer plays a very important role, connecting the task management layer and the task execution layer. Formation tasks of USVs can be divided into forming a designated formation, sailing according to a fixed formation, and formation transformation. Therefore, the path planning layer also needs corresponding path planning strategies; that is, the

path planning layer needs to solve three types of planning problems: formation generation, formation transformation, and formation navigation. The formation path planning of USVs is based on the path planning of a single USV.

The aim of path planning is to find a feasible path in a specific environment. This path begins at the starting point and ends at the target point [7]. The path planning problem can be regarded as a multiobjective optimization problem [8]. The trajectory should be optimized in distance, time, energy consumption, risk cost, and other aspects. In addition, the smoothness of the trajectory and whether it meets the kinetic constraints of the mobile robot cannot be ignored. The solution methods of multiobjective optimization problems are mainly of two categories: traditional optimization methods and intelligent optimization methods. Traditional optimization algorithms include weighting method [9], constraint method [10], and linear programming method [11], which essentially transform the multiobjective function into single-objective function and achieve the solution of multiobjective function by using the single-objective optimization method. Intelligent optimization algorithms include genetic algorithms [12], artificial neural networks [13,14], and so on. A good USV formation path planning method needs to meet the following indicators [15]:

- Rationality: Any path of return is reasonable, or any path is feasible for formation movement.
- Completeness: If, objectively, there is a collision-free path from the starting point to the endpoint, the algorithm can find it; if no path is available, a planning failure is reported.
- Optimality: The resulting path planned by the algorithm is optimal on some measures (such as time, distance, energy consumption, etc.).
- Real-time: The complexity of the planning algorithm (time requirements, storage requirements, etc.) can meet the needs of USV movement.
- Satisfy constraint: Supporting the nonintegrity constraint of USV movement.

Much research has been carried out on formation path planning methods for unmanned aerial vehicles (UAVs), unmanned ground vehicles (UGVs), and mobile robots čitezuo2016coverage, qiu2014receding, vicmudo2014path. In recent years, some research on path planning of USV formation has emerged. Tam [19] proposed a multiboat coordinated path planning algorithm that conforms to the COLREGS rule. The simulation results show that the output of the algorithm is effective and consistent in various traffic scenarios. Liu [6] proposed a constrained fast-marching method to address the dynamic problem in path planning of USV formation. The simulation results showed that collision-free paths can be generated for formations for complex, practical, and for both static and dynamic environments. Sang [20] presented a hybrid path planning algorithm based on improved A* and multiple subtarget artificial potential field (MTAPF). The MTAPF belongs to the local path planning algorithm, which refers to the global optimal path generated by an improved heuristic A* algorithm. The simulation results showed that the algorithm can greatly reduce the probability that USV will fall into the local minimum and help USV to move out of the local minimum by switching target points. Ouyang [21] proposed an algorithm for USV formation path planning based on an improved rapidly-exploring random tree (RRT) algorithm for global path planning and obstacle avoidance. This research showed that the improved algorithm has such advantages as high efficiency, good stabilization, and high-quality planning paths. Wang [22] proposed a path planning algorithm for USV formation based on the variable fast-marching method. The method regards the feasible area as anisotropy and considers that the safety of a point is linearly related to the distance between the point and the obstacle. The method was experimentally proven to have good real-time performance. In addition, the traditional A* algorithm [23], particle swarm optimization [24], genetic algorithm [25], and ant colony algorithm [26] are also used to solve the problem of multirobot formation path planning, but they are seldom applied to USV formation path planning.

Path planning can be divided into three stages: route planning, trajectory planning, and motion planning, according to whether the kinematics and dynamics of the USV are

considered in the planning process [27]. In the route planning stage, the USV is regarded as a particle model. The shape, dynamics, and kinematics constraints of the USV are not considered. In the trajectory planning stage, some constraints, such as the shape of the USV and the minimum turning radius, are considered in the planning. At the stage of motion planning, considering the kinetic model of USV, the path suitable for USV tracking is planned. However, most of the previous studies [28–31] are only at the route planning level and lack the consideration of the USV kinetic model. The path may not satisfy the nonintegrity constraints of the USV. In addition, formation problems include formation generation, formation transformation, and formation navigation. Most of the previous studies focused on formation navigation, but few systematically studied the three problems. Therefore, this paper focuses on solving the path planning problem of USVs considering the kinetic model and provides corresponding solutions to formation generation, formation navigation, and formation transformation of USVs.

1.2. The Main Work and Contributions of the Article

1. The article proposes a path planning algorithm for USVs based on improved A* algorithm. The output path of the algorithm is composed of the minimum trajectory elements generated by the dynamic model, which can ensure that the trajectory fully conforms to the kinetic constraints of USV, and the path can be directly tracked by USV without subsequent processing.
2. The article proposes a heuristic value map construction method. A* algorithm can query the heuristic value map to obtain the appropriate heuristic value during search, which can significantly speed up the algorithm search speed.
3. For the first time, the article divides USV formation path planning into formation navigation path planning and formation reconfiguration path planning. We give solutions for these two kinds of path planning based on the above improved A* algorithm.
4. In this paper, model identification experiments are conducted for the SL900 USV, and simulation experiments are conducted for the above algorithm based on the experimental results to verify the rationality of the algorithm.

1.3. The Main Structure of the Article

Section 1 gives the introduction to the article. Section 2 introduces an improved A* algorithm based on the minimum trajectory elements for a single USV. In Section 3, the mathematical model of USV formation based on the virtual structure method is established, and the path planning algorithm of USV formation navigation and formation reconfiguration is given. In Section 4, we conduct a model identification experiment of USV. The above algorithms are verified by simulation experiments. In Section 5, we discuss the advantages and disadvantages of the proposed method. Section 6 summarizes the paper and introduces future work.

2. The Path Planning Method for a Single Unmanned Surface Vehicle

An improved A* path planning algorithm for USV based on minimum trajectory elements is proposed in this section. Compared with the traditional A* algorithm, the path generated by this method is guaranteed to comply with the USV kinetic constraints. The method is described in detail below.

2.1. The Introduction of Traditional A Algorithm*

A* algorithm is a search method to find the path with the least cost in the grid map, where the cost can be distance, risk cost, etc. It was first proposed by Hart in 1968 [32]. This method combines the advantages of the Dijkstras algorithm [33] and the best-first algorithm [34]. Based on an evaluation function, the A* algorithm can not only improve the

efficiency of heuristic search, but it also ensures finding an optimal path. The evaluation function is expressed in Equation (1).

$$f(n) = g(n) + h(n) \tag{1}$$

where $f(n)$ is the estimated cost from the start point S to the target point T through the node n; $g(n)$ is the actual cost from start point S to the node n; $h(n)$ is the estimated cost of reaching the target point T from the node n, $h(n)$, also known as heuristic values.

Different heuristic value functions have a great influence on the speed and accuracy of path search by the A* algorithm [35]. If the heuristic value of a point is exactly equal to the actual cost of the point to the target point, the A* algorithm can ensure that the optimal path can be found at a fast speed. If the heuristic value of a point is greater than the actual cost of the point to the target point, A* can find the path to the target point at a faster speed, but it does not guarantee that the path is optimal. If the heuristic value of a point is less than the actual cost of the point to the target point, the A* algorithm can ensure that the optimal path to the target point can be found, and the search speed will be correspondingly slower, but also faster than Dijkstras algorithm. In the practical application process, it is the best choice to choose the heuristic value function equal to the actual cost, but it is difficult to obtain this function. In general, the optimal path will be guaranteed first, and the function whose heuristic value is slightly less than the actual cost will be selected.

The pseudocode of the traditional A* algorithm is shown in Algorithm 1, where S is the start point, T is the target point, $OPENLIST$ is the set of nodes to be searched, $CLOSELIST$ is the set of searched nodes, $OBSTACLESLIST$ is a collection of obstacles, $G(n)$ represents the set of neighbor nodes of node n, and $C(n, n')$ represents the actual cost from node n to node n'. If "Path is found" is displayed after the algorithm runs, the optimal path is obtained by connecting the node from the target point to its parent until the start node.

Algorithm 1 Traditional A* algorithm.

Input: start point S, target point T and environment map M
Output: Whether there is a path from the start point to the target point.

Add the start point S to $OPENLIST$.
while $OPENLIST \neq \emptyset$ **do**
  Take the point n with the smallest value of the evaluation function $f(n)$ from $OPENLIST$.
  Delete n from $OPENLIST$ and add it to $CLOSELIST$.
  if $n = T$ **then**
    return "Path is found."
  else
    Get the set $G(n)$ composed of all neighbor points of n.
    for all point $n' \in G(n)$ **do**
      if $n' \notin CLOSELIST$ and $n' \notin OBSTACLESLIST$ **then**
        if $n' \in OPENLIST$ **then**
          if $g(n') > g(n) + C(n, n')$ **then**
            Set the parent of n' in $OPENLIST$ to be n and $g(n') = g(n) + C(n, n')$.
          end if
        else
          Insert n' into $OPENLIST$. Set the parent of n' is n, $g(n') = g(n) + C(n, n')$, $f(n') = g(n') + h(n')$.
        end if
      end if
    end for
  end if
end while
return "The path does not exist."

As a classical graph search algorithm, the A* algorithm is widely used in the shortest path search problem. Although the A* algorithm can ensure the shortest path, the continuity and smoothness of the path are poor, so it is not suitable for mobile robots to follow, especially mobile robots with non-holonomic constraints. Therefore, the application of the A* algorithm in the scenario of USV formation path planning still needs some improvements, which will be introduced in detail below.

2.2. Improved A* Algorithm

In order to plan a trajectory that conforms to the kinetic constraints of the USV, we need to consider more constraints. On the basis of only considering the two-dimensional space of position, we additionally consider the yaw of USV, so that the state space of USV has to be increased to three-dimensional space.

There are infinite state points on the high-dimensional continuous space, and it is not feasible to search directly on the space. Before the search, it is usually necessary to discretize the state space of the USV, and transform the arbitrary state search problem into a specific state search problem to reduce the complexity of the search algorithm.

In this paper, a raster map is used to discretize the USV state space. As shown in Figure 2, USVs are represented by vector points with yaw information on the raster map. Each grid is a cell, and the USV can be in any position on each grid. In order to simplify the state space, discrete sampling is conducted on the yaw of the USV within the range of $[0, 2\pi)$. According to the yaw information obtained by sampling, the state of the USV in the above grid cells is restricted; specifically, there can only be one state point with the same yaw information in each grid cell. In addition, the concept of neighborhood is defined. For a certain state of USV, the neighborhood of this state refers to the set of all state points with the same yaw information in the same grid. The neighborhood of a state is denoted by $Neighborhood(P)$.

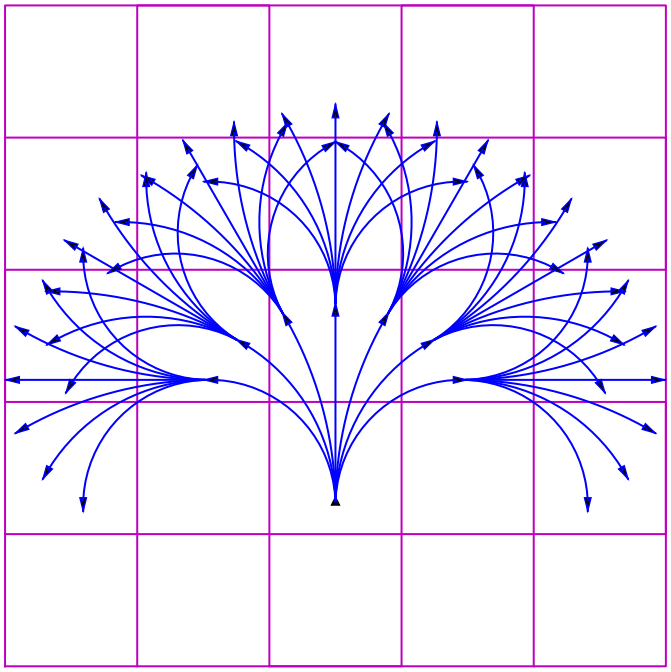

Figure 2. Representation of USV status and trajectory on raster map.

In addition to the above improvements, we also propose heuristic value maps and a minimum trajectory elements method. The heuristic value map records the appropriate heuristic value, which is used to speed up the path search; the minimum trajectory elements are used to connect the state points of the USV and ensure that the path conforms to the kinetic constraints, and these two methods are detailed in Sections 2.3 and 2.4, respectively.

The improved A* algorithm proposed in this paper, which is suitable for the path planning of USV, is shown in Algorithm 2, where $C(\zeta_{Pi})$ represents the cost of ζ_{Pi}; $Par(P)$ represents the last state of state P and the minimum trajectory element ζ_{Pi} between the two states. That is, $Par(P) = P, \zeta_{Pi}$.

Algorithm 2 Improved A* algorithm for USV path planning.

Input: start point $S(x, y, \varphi)$, target point $T(x, y, \varphi)$ and environment map M.
Output: The optimal path L.

Establish heuristic value map HM according to target state $T(x, y, \varphi)$ and Algorithm 3.
Clear sets $OPENLIST$,$CLOSELIST$. Add the initial state $S(x, y, \varphi)$ to $OPENLIST$.
Query HM to get $H(S)$ of state S. Set $F(S) = C(S, S) + H(S) = H(S)$.
while $OPENLIST \neq \emptyset$ **do**
  Pick the waypoint P with the smallest value of $F(P)$ from $OPENLIST$.
  Delete P form $OPENLIST$. Add P to $CLOSELIST$.
  if $P \in Neighborhood(T)$ **then**
    Take the state P that belongs to $Neighborhood(T)$ from $CLOSELIST$. Set the optimal path of USV $L = \emptyset$.
    while $P \neq S$ **do**
      Update $P, \zeta_{Pi} = Par(P)$. Add ζ_{Pi} to L.
    end while
    Reverse the order of the minimum trajectory elements in L.
    return L
  end if
  Generate the minimum trajectory elements set $U_P = \{\zeta_{P1}, \zeta_{P2}, \cdots \zeta_{Pn}\}$ from the P. The state P superimposes the U_P to reach the state set $P' = \{P'_1, P'_2, \cdots P'_n\}$
  for all $P'_i \in P'$ **do**
    Query HM to get $H(P'_i)$ of state P'_i. Set $C(S, P'_i) = C(S, P) + C(\zeta_{Pi})$.
    if there is a state $P''_i \in Neighborhood(P'_i)$ in $CLOSELIST$ and $C(S, P''_i) > C(S, P'_i)$ **then**
      Deletes the state P''_i from $CLOSELIST$.
    end if
    if there is a state $P'''_i \in Neighborhood(P'_i)$ in $OPENLIST$ and $C(S, P'''_i) > C(S, P'_i)$ **then**
      Deletes the state P'''_i from $OPENLIST$.
    end if
    if there is no state point in $OPENLIST$ and $CLOSELIST$ that belong to $Neighborhood(P'_i)$ **then**
      Add P'_i to $OPENLIST$. Set $F(P'_i) = C(S, P'_i) + H(P'_i)$, $Par(P'_i) = P, \zeta_{Pi}$.
    end if
  end for
end while
return $\emptyset$

2.3. The Method of Building Heuristic Value Map

As described in Section 2.1, the heuristic value affects the search speed and the quality of the obtained path of the A* algorithm. Some researchers proposed various heuristic value design methods for different robots [36,37]. Euclidean distance and Manhattan distance are currently commonly used heuristic value functions. Euclidean distance is the straight-line distance between two points. This distance must be less than or equal to the actual cost between the two points, so it has optimality. The Manhattan distance is the sum of the absolute values of the difference between the coordinates of two points. This distance must be greater than or equal to the actual cost between the two points, so it may not be optimal.

The previous analysis of Manhattan distance and Euclidean distance does not take into account obstacles in the environment and the yaw constraints of USV, both of which may increase the actual cost. This causes the Euclidean distance to become smaller compared to the actual cost of the path and prolongs the algorithm search time. Therefore, it is necessary

to find a heuristic value calculation method that considers the obstacles in the environment and the yaw constraints of USV.

Inspired by the literature [38], we design a method of building a heuristic map based on Dijkstras algorithm to solve the above problems. With the help of Dijkstras algorithm, the reverse search is carried out from the target point. In the search process, the neighbor grid and the different costs of the USV from the current grid to the neighbor grid are set according to the current yaw of the USV to achieve the purpose of avoiding obstacles and restraint of the USV. For the specific settings, see Figure 3, where the red arrow represents the current state; the black arrow represents the next state. The black square is the obstacle; the cross is the neighbor grid not considered; the cost of reaching the next state is recorded in the lower right corner of the grid, and γ is the unit cost. In the process of constructing the heuristic value map, four effective directions of the USV are set, and the five neighboring grids are the effective next state (as shown in (a), (b), (c), and (d) in Figure 3). (I), (II), (III), (IV), and (V) take the state in (a) as an example to show the selection rule when there are obstacles in the neighbor grid. The minimum path cost to reach each raster is used as the heuristic value of the waypoint in that raster. The heuristic value is filled into each raster. The heuristic value in the obstacle raster is set to positive infinity. Thus, we obtain a map recording the heuristic value at each raster.

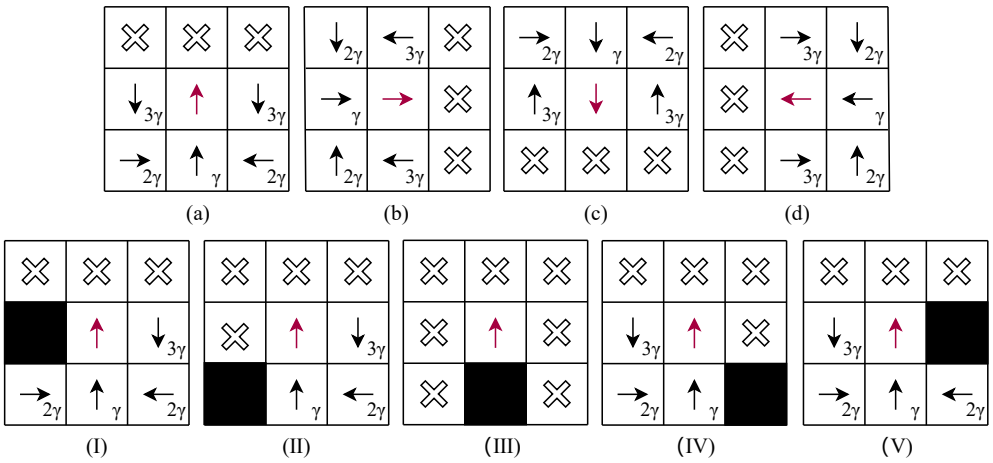

Figure 3. Selection rules of neighbor nodes in the process of building heuristic value map ((**a**–**d**) show the neighbor nodes selection rules for different initial states; (**I**–**V**) take the state in (**a**) as an example to show the selection rule when there are obstacles in the neighbor grid).

When A* algorithm searches the path, the value recorded on the raster is directly queried as the heuristic value. This method makes the heuristic value closer to the actual cost, which can improve the search speed of the algorithm and make it have better real-time performance. The establishment steps of the heuristic value map are shown in Algorithm 3. An example of a heuristic value map is shown in Figure 4, where the blue arrow indicates the target state of the USV. The minimum cost from the target raster to that raster is stored in each raster; the raster that stores the value ∞ in the figure represents an obstacle or map boundary. Setting the heuristic value to infinity can avoid obstacles in path planning.

Algorithm 3 Heuristic map building algorithm.

Input: target point $T(x, y, \varphi)$ and environment map M
Output: heuristic map HM

Copy the grid environment map M as the initial heuristic map HM.
Fill each grid with the heuristic value ∞, empty sets $OPENLIST$ and $CLOSELIST$.
Add target point $T(x, y, \varphi)$ to set $OPENLIST$, $C(T, T) = 0$.
while $OPENLIST \neq \emptyset$ **do**
  Pick the waypoint P with the smallest value of $C(T, P)$ from $OPENLIST$.
  Delete P form $OPENLIST$. Add P to $CLOSELIST$. Update the heuristic value in the grid where P is to $C(T, P)$.
  Obtain the waypoint set $G(P)$ in all neighbor grids of the grid where P is. And delete the waypoints in the set $CLOSELIST$ or $UNREACHABLE$ from $G(P)$.
  for all point $P' \in G(P)$ **do**
    if $P' \notin OPENLIST$ **then**
      Add P' to $OPENLIST$. Set $C(T, P') = C(T, P) + C(P, P')$.
    end if
    if $P' \in OPENLIST$ and $C(T, P') > C(T, P) + C(P, P')$ **then**
      Set $C(T, P') = C(T, P) + C(P, P')$.
    end if
  end for
end while
return HM

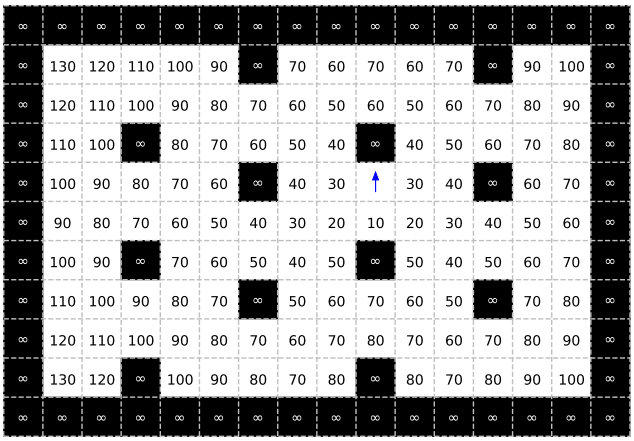

Figure 4. An example of a heuristic value map ($\gamma = 10$).

2.4. The Generation Method of Minimum Trajectory Elements

The traditional A* algorithm directly connects the adjacent state points to generate the path. The path generated by this method lacks continuity and smoothness, which does not meet the kinetic constraints of USV. Inspired by the literature [39], this paper proposes a method to generate the minimum trajectory elements according to the kinetic model of USV to solve the above problem. The minimum trajectory elements are used to connect two path points searched by the algorithm. The path composed of the minimum trajectory elements can satisfy the kinetic constraints of the USV.

In order to reduce the complexity of the research, it is usually only necessary to study the motion of the USV in the horizontal plane, that is, the three degrees of freedom of forward motion, transverse drift, and yaw are considered. The three degree of freedom motion model of the USV is shown in Figure 5. The kinetic model of USV is complex, and

there are many simplified forms, but the methods are generally similar. This paper will generate the minimum trajectory elements based on the model [40,41] in Equation (2).

$$\begin{cases} \dot{x} = u\cos\varphi - v\sin\varphi \\ \dot{y} = u\sin\varphi + v\cos\varphi \\ \dot{\varphi} = r \\ \dot{u} = a_u u + b_u T \\ \dot{r} = c_r r + d_r \delta \end{cases} \tag{2}$$

where x and y represent the position of the USV in the inertial coordinate system; $\dot{x}$ and $\dot{y}$ represent the velocity of the USV in the inertial coordinate system; u represents the forward speed of the USV; $\dot{u}$ represents the forward acceleration of the USV; v represents the transverse speed of the USV; φ represents the yaw of the USV; r and $\dot{\varphi}$ are the yaw velocity of the USV; $\dot{r}$ represents the yaw acceleration of the USV; T represents the thrust of the USV; δ represents the rudder of the USV. a_u, b_u, c_r, and d_r are coefficients, which can be determined through identification experiments.

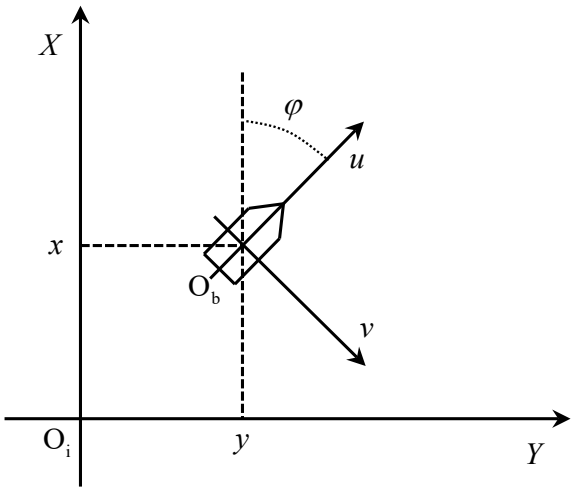

Figure 5. Three degrees of freedom motion model of USV.

The system state variable of the USV is defined as $\boldsymbol{x} = \begin{bmatrix} x & y & \varphi & u & v & r \end{bmatrix}^{\mathrm{T}}$. The control input is $\boldsymbol{u} = \begin{bmatrix} T & \delta \end{bmatrix}^{\mathrm{T}}$, and the system output is $\boldsymbol{y} = \begin{bmatrix} x & y & \varphi \end{bmatrix}^{\mathrm{T}}$. Thus, the state space model of the USV is:

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u}) = \begin{bmatrix} u \cdot \cos\varphi - v \cdot \sin\varphi \\ u \cdot \sin\varphi + v \cdot \cos\varphi \\ r \\ a_u u + b_u T \\ 0 \\ c_r r + d_r \delta \end{bmatrix} \tag{3}$$

$$\boldsymbol{y} = P\boldsymbol{x} \tag{4}$$

where $P = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$.

If the above Equation (3) is expanded by Taylor at any point $(\boldsymbol{x}_0, \boldsymbol{u}_0)$ and the first-order term is retained, we can obtain:

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u}) = f(\boldsymbol{x}_0, \boldsymbol{u}_0) + \left.\frac{\partial f}{\partial \boldsymbol{x}}\right|_{\substack{\boldsymbol{x}=\boldsymbol{x}_0 \\ \boldsymbol{u}=\boldsymbol{u}_0}} (\boldsymbol{x} - \boldsymbol{x}_0) + \left.\frac{\partial f}{\partial \boldsymbol{u}}\right|_{\substack{\boldsymbol{x}=\boldsymbol{x}_0 \\ \boldsymbol{u}=\boldsymbol{u}_0}} (\boldsymbol{u} - \boldsymbol{u}_0) \tag{5}$$

By combining Equations (3) and (5), we can obtain:

$$\dot{\tilde{\boldsymbol{x}}} = A\tilde{\boldsymbol{x}} + B\tilde{\boldsymbol{u}} \tag{6}$$

where $\tilde{\boldsymbol{x}}$ is the deviation between the actual system and the reference system, $\tilde{\boldsymbol{x}} = \boldsymbol{x} - \boldsymbol{x}_0$; $\tilde{\boldsymbol{u}}$ is the deviation between the control quantity of the actual system and the reference system of the USV, and $\tilde{\boldsymbol{u}} = \boldsymbol{u} - \boldsymbol{u}_0$; A is the Jacobian matrix of f relative to $\boldsymbol{x}$; B is the Jacobian of f relative to $\boldsymbol{u}$; and there are:

$$\begin{cases} A = \left.\frac{\partial f}{\partial \boldsymbol{x}}\right|_{\substack{\boldsymbol{x}=\boldsymbol{x}_0 \\ \boldsymbol{u}=\boldsymbol{u}_0}} = \begin{bmatrix} 0 & 0 & -u\sin\varphi - v\cos\varphi & \cos\varphi & -\sin\varphi & 0 \\ 0 & 0 & u\cos\varphi - v\sin\varphi & \sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & a_u & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & c_r \end{bmatrix} \\ B = \left.\frac{\partial f}{\partial \boldsymbol{u}}\right|_{\substack{\boldsymbol{x}=\boldsymbol{x}_0 \\ \boldsymbol{u}=\boldsymbol{u}_0}} = \begin{bmatrix} 0 & 0 & 0 & b_u & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & d_r \end{bmatrix}^{\mathrm{T}} \end{cases}$$

Equation (6) was discretized by Euler's method to obtain a state space model of the USV in discrete form, as shown in Equation (7).

$$\begin{cases} \tilde{\boldsymbol{x}}(k+1) = A_k\tilde{\boldsymbol{x}}(k) + B_k\tilde{\boldsymbol{u}}(k) \\ \tilde{\boldsymbol{y}}(k) = P\tilde{\boldsymbol{x}}(k) \end{cases} \tag{7}$$

where $A_k = A \cdot \mathrm{d}t + I$; $B_k = B \cdot \mathrm{d}t$; $\mathrm{d}t$ is the time step; I is the identity matrix; k is any time.

According to Equation (7), after applying certain control input to the USV, we can obtain the next determined motion trajectory of the USV. Therefore, based on the current state of the USV, we can obtain a group of different motion trajectories of the USV after a period of time by giving a group of different control inputs, which is the set of minimum trajectory elements. In order to ensure that the path assembled by the minimum trajectory elements is smooth and the velocity on the path is continuous, the generation process of the minimum trajectory elements set meets the following conditions:

1. Thrust T is a constant, and rudder δ is sampled uniformly and discretely between the maximum and minimum to obtain $\{\delta_1, \delta_2, \cdots, \delta_n\}$, which is combined to obtain control input $\boldsymbol{u_1} = \{(T, \delta_1), (T, \delta_2), \cdots, (T, \delta_n)\}$,$\boldsymbol{u_2} = \{(T, 0), (T, 0), \cdots\cdots, (T, 0)\}$, from which a set of minimum trajectory elements consisting of n minimum trajectory element is obtained.
2. Each minimum trajectory element is divided into two parts. The first part applies the control input $\boldsymbol{u_1}$, and the second part applies the control input $\boldsymbol{u_2}$ to ensure that the u, v, and r of the USV at both ends of the minimum trajectory element are the same.
3. The operation time of control input is the same for each group.

An example of the set of minimum trajectory elements is shown in Figures 6 and 7 shows the change in yaw velocity and corresponding rudder of one of the minimum trajectory elements.

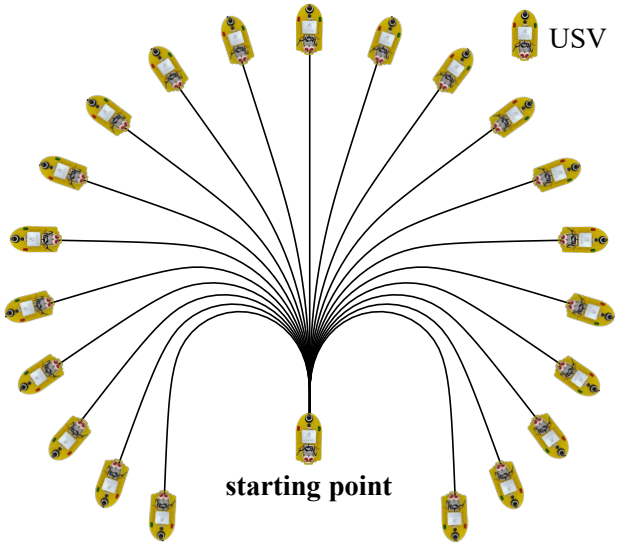

Figure 6. Minimum trajectory elements generated by different rudder loads and given thrust.

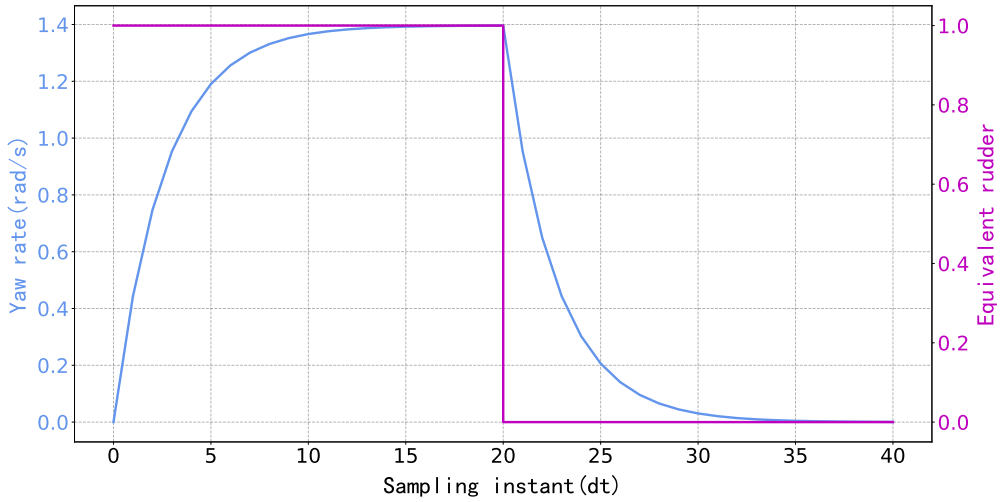

Figure 7. The change curve of yaw velocity and rudder in minimum trajectory element.

3. The Path Planning Method for Unmanned Surface Vehicles Formation

We classify path planning for USV formation into two types, which are USV formation navigation and USV formation reconfiguration, where USV formation reconfiguration includes formation generation and formation transformation. Both path planning processes are based on the path planning of an individual USV. The methods for both formation path planning are described in detail below.

3.1. Mathematical Model of USV Formation

In this paper, the virtual structure method will be used to organize the formation of the USV. The virtual structure method requires the virtual rigid body to be determined in advance according to the formation requirements of the USV. The virtual rigid body is an abstraction of the rigid whole composed of multiple USVs [42]. The coordinate system of the USV formation is established as shown in Figure 8, where $XO_{\text{inertial}}Y$ is the inertial coordinate system, $xO_{\text{rigid}}y$ is the rigid body coordinate system (RBCS), and $uO_{\text{body}}v$ is the hull coordinate system. The USV constitutes a certain spatial distribution under the rigid

body coordinate system, which is the formation of the USV. The position of each USV under the rigid body coordinate system $xO_{\text{rigid}}y$ at the moment t is expressed as Equation (8).

$$Ps_{\text{rigid}}(t) = \left\{ P_{\text{rigid},1}(t), P_{\text{rigid},2}(t), \cdots, P_{\text{rigid},n}(t) \right\} \tag{8}$$

where $P_{\text{rigid},i}(t)$ denotes the position $\left[x_{\text{rigid},i}, y_{\text{rigid},i} \right]^{\text{T}}$ of the USV i in the rigid body coordinate system. When $P_{\text{rigid},i}(t)$ is a constant function, a fixed virtual rigid body structure is formed and the formation remains fixed. When $P_{\text{rigid},i}(t)$ is a time-varying function, formation transformation can be realized.

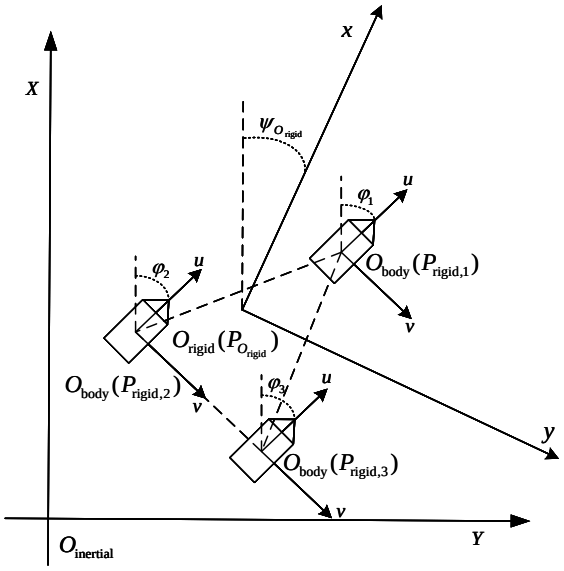

Figure 8. The coordinate system of USV formation.

The position and posture of the virtual rigid body in the inertial coordinate system are $P_{O_{\text{rigid}}}(t)$ and $\psi_{O_{\text{rigid}}}(t)$, respectively, where the posture of the virtual rigid body in the inertial coordinate system refers to the included angle between the two coordinate systems. According to the position and posture of the virtual rigid body and the position of each USV in the virtual rigid body coordinate system, the expression of the position of each USV under the inertial coordinate system can be calculated as Equation (9).

$$P_{\text{inertial},i}(t) = P_{O_{\text{rigid}}}(t) + R_{\text{rigid2Inertial}}(t) \cdot P_{\text{rigid},i}(t) \tag{9}$$

where $P_{\text{inertial},i}(t)$ is the position of the USV i in the inertial coordinate system at moment t; $R_{\text{rigid2Inertial}}(t)$ is the transformation matrix from the rigid body coordinate system to the inertial coordinate system, as shown in Equation (10).

$$R_{\text{rigid2inertial}}(t) = \begin{bmatrix} \cos\left(\psi_{O_{\text{rigid}}}(t)\right) & -\sin\left(\psi_{O_{\text{rigid}}}(t)\right) \\ \sin\left(\psi_{O_{\text{rigid}}}(t)\right) & \cos\left(\psi_{O_{\text{rigid}}}(t)\right) \end{bmatrix} \tag{10}$$

3.2. The Path Planning for USV Formation Navigation

During the formation navigation, the formation of USVs remains unchanged, that is, the position of each USV in the virtual rigid body coordinate system does not change with time and the virtual structure is fixed. The motion trajectory of any point on the virtual rigid body is parallel to the motion trajectory of the virtual rigid body coordinate system. If the virtual rigid body coordinate system moves along the minimum trajectory element, any point on the virtual rigid body moves along the minimum trajectory element. Therefore, during the USV formation navigation, the path planning process can only plan the path for

the virtual rigid body coordinate system considering the kinetic constraints of the USV. In addition, the formation of the USV needs to be considered in the planning process. The current state and next state of each USV can be calculated according to the current state and next state of the virtual rigid body coordinate system during the search by the improved A* algorithm, and the minimum trajectory element between the states is judged as to whether there is a collision with an obstacle. The mathematical model of path planning for USV formation navigation is shown in Equation (11).

$$\begin{cases} P_{t_i} = \left(P_{O_{\text{rigid}}}(t_i), \psi_{O_{\text{rigid}}}(t_i)\right) \\ \zeta_{t_i} = f(P_{t_i}, \mathrm{M}_{\text{kinetic}}, \mathrm{d}t) \\ l_{\text{rigid}} = \{\zeta_{\mathrm{t}_1}, \zeta_{\mathrm{t}_2}, \cdots, \zeta_{t_n}\} \\ \zeta_{t_i,j} = \zeta_{t_i} + P_{\text{rigid},j}(t_i) \\ l_j = \{\zeta_{t_1,j}, \zeta_{t_2,j}, \cdots, \zeta_{t_n,j}\} \end{cases} \tag{11}$$

where P_{t_i} represents the state of the virtual rigid body coordinate system at moment t_i in the inertial coordinate system; ζ_{t_i} represents the minimum trajectory element of the virtual rigid body coordinate system at moments t_i to t_{i+1}, which is generated by P_{t_i}, the kinetic model $\mathrm{M}_{\text{kinetic}}$ of USV (see Equation (7)), and time interval dt (see Section 2.4 for the specific generation method); l_{rigid} represents the optimal trajectory of the virtual rigid body coordinate system generated by the improved A* algorithm from the initial state to the target state in the inertial coordinate system, which is assembled by the minimum trajectory elements and conforms to the kinetic constraints of the USV; $\zeta_{t_i,j}$ represents the minimum trajectory element of USV j at time t_i to t_{i+1}, which can be calculated from the minimum trajectory element ζ_{t_i} of the virtual rigid body coordinate system and virtual structure $Ps_{\text{rigid}}(t_i)$ at the time t_i. l_i represents the optimal trajectory of the USV j from the initial state to the target state in the inertial coordinate system.

The path planning algorithm for USV formation navigation is shown in Algorithm 4.

3.3. The Path Planning for USV Formation Reconfiguration

USV formation reconfiguration path planning can be achieved by making each USV individually planned to a target point. It differs from the individual USV path planning in that the formation reconfiguration path planning needs to consider the collision avoidance between USVs. During the reconfiguration process, the positions of the USV clusters in the inertial coordinate system should not be equal for any given time. In the actual path searching, the USV cluster shares a real-time updated environment map. The USVs in the cluster plan their optimal paths from the current state to the target state one by one. After each USV finds the optimal path, the position of the USV is added to the environment map with the corresponding time stamp according to the time stamp on the path information, and the position on the environment map is equivalent to the impassable area for other USVs at that moment. When the last USV in the cluster finds the optimal path to the target point, the formation reconfiguration path planning is completed. The algorithm of path planning for formation reconfiguration is shown in Algorithm 5, and the mathematical model of formation reconfiguration path planning is shown in Equation (12).

$$\begin{cases} Ps_{\text{inertail}}(t_0) = \left\{P_{\text{inertail},1}(t_0), P_{\text{inertail},2}(t_0), \cdots, P_{\text{inertail},n}(t_0)\right\} \\ P_{\text{Inertial,i}}\left(t_{\text{target}}\right) = P_{O_{\text{rigid}}}\left(t_{\text{target}}\right) + R_{\text{rigid2Inertial}}\left(t_{\text{target}}\right) P_{\text{rigid,i}}\left(t_{\text{target}}\right) \\ L = \{l_1, l_2, \cdots, l_n\} \\ l_i = \left\{\zeta_{\text{i,t}_1}, \zeta_{\text{i,t}_2}, \cdots, \zeta_{\text{i,t}_{\text{target}}}\right\} \\ \text{for} \forall t_i \text{and} \forall \zeta_{\text{j,t}_\text{i}} \zeta_{\text{m,t}_\text{i}} \text{have} \zeta_{\text{j,t}_\text{i}} \cap \zeta_{\text{m,t}_\text{i}} = \varnothing \end{cases} \tag{12}$$

where $Ps_{\text{inertail}}(t_0)$ represents the state given to the USV in the inertial coordinate system before formation reconfiguration; $P_{\text{inertial},i}\left(t_{\text{target}}\right)$ is the target point of the USV i in the inertial coordinate system, L is the optimal path set of each USV during formation reconfiguration, and l_i is the optimal path of the USV i, which is assembled by the minimum

trajectory elements. The minimum trajectory elements of any USV do not intersect at the same time.

Algorithm 4 The path planning algorithm for the sailing of USVs in fixed formation.

Input: starting point of RBCS $P_S(x_S, y_S, \psi_S)$, target point of RBCS $P_T(x_T, y_T, \psi_T)$, environment map M and virtual structure VS.
Output: The optimal trajectory set $L = \{l_1, l_2, \cdots l_n\}$.

Establish heuristic value map HM according to target state P_T and Algorithm 3.
Clear sets $OPENLIST$ and $CLOSELIST$. Add the initial state P_S to $OPENLIST$.
Query HM to get $H(P_S)$ of state P_S. Set $F(P_S) = C(P_S, P_S) + H(P_S) = H(P_S)$.
while $OPENLIST \neq \emptyset$ **do**
  Pick the waypoint P with the smallest value of $F(P)$ from $OPENLIST$.
  Delete P form $OPENLIST$. Add P to $CLOSELIST$.
  if $P \in Neighborhood(P_T)$ **then**
    Take the state P that belongs to $Neighborhood(P_T)$ from $CLOSELIST$. Set the optimal path of RBCS $l_{\text{rigid}} = \emptyset$.
    while $P \neq P_S$ **do**
      Update $P, \zeta_P = Par(P)$. Add ζ_P to l_{rigid}.
    end while
    Reverse the order of the minimum trajectory elements in l_{rigid}.
    According to the sequence and time interval t of the minimum trajectory elements on the l_{rigid}, time constraint is added to each minimum trajectory element to obtain $l_{\text{rigid}} = \{\zeta_{t_1}, \zeta_{t_2}, \cdots, \zeta_{t_n}\}$.
    According to Equation (11) and l_{rigid}, generating the trajectory l_i of USV i.
    return $L = \{l_1, l_2, \cdots l_n\}$
  end if
  Generate the minimum trajectory elements set $U_P = \{\zeta_{P1}, \zeta_{P2}, \cdots \zeta_{Pn}\}$ from the P.
  According to Equation (11) and U_P, generating the minimum trajectory elements set $U_{P,j} = \{\zeta_{P1,j}, \zeta_{P2,j}, \cdots \zeta_{Pn,j}\}$ of USV j.
  if there have $\zeta_{Pi,j}$ or ζ_{Pi} collides with obstacle **then**
    Delete the $\zeta_{Pi,j}$ or ζ_{Pi} from U_P and $U_{P,j}$.
  end if
  The state P superimposes the minimum trajectory elements set U_P to reach the state set $P' = \{P'_1, P'_2, \cdots P'_n\}$.
  for all $P'_i \in P'$ **do**
    Query HM to get $H(P'_i)$ of state P'_i. Set $C(S, P'_i) = C(S, P) + C(\zeta_{Pi})$.
    if there is a state $P''_i \in Neighborhood(P'_i)$ in $CLOSELIST$ and $C(S, P''_i) > C(S, P'_i)$ **then**
      Delete the state P''_i from $CLOSELIST$.
    end if
    if there is a state $P'''_i \in Neighborhood(P'_i)$ in $OPENLIST$ and $C(S, P'''_i) > C(S, P'_i)$ **then**
      Delete the state P'''_i from $OPENLIST$.
    end if
    if there is no state point in $OPENLIST$ and $CLOSELIST$ that belong to $Neighborhood(P'_i)$ **then**
      Add P'_i to $OPENLIST$. Set $F(P'_i) = C(S, P'_i) + H(P'_i)$, $Par(P'_i) = P, \zeta_{Pi}$.
    end if
  end for
end while
return $\emptyset$

Algorithm 5 The algorithm of path planning for formation reconfiguration.

Input: Initial states $Ps_{\text{inertail}}(t_0)$, target states $Ps_{\text{inertail}}(t_{\text{target}})$ of USV, environment map M.
Output: The optimal trajectory set $L = \{l_1, l_2, \cdots l_n\}$.

```
Set the optimal trajectory of RBCS L = ∅
for all P_inertail,i(t_0), P_inertial,i(t_target) ∈ Ps_inertail(t_0), Ps_inertail(t_target) do
    Input P_inertail,i(t_0), P_inertial,i(t_target) and M to run Algorithm 2 to obtain the optimal trajectory l_i of USV i.
    According to the sequence and time interval of the minimum trajectory elements on the l_i, time constraint is added to each minimum trajectory element to obtain l_i = {ζ_{t_1,i}, ζ_{t_2,i}, ···, ζ_{t_n,i}}.
    for all ζ_{t_j,i} ∈ l_i do
        Add obstacles to ζ_{t_j,i}'s position on the environment map M at t_j time.
    end for
    Add l_i to L.
end for
return L
```

4. Algorithm Validation

We carry out model identification experiments of USVs and design two types of simulation experiments to verify the effectiveness of the proposed USV formation path planning algorithm. The first experiment is the path planning of a single USV. The results are compared with the conventional A* algorithm. The second one is the path planning of USV formation, including three scenarios of formation navigation, formation reconfiguration, and combination. The algorithms for these experiments were implemented in Python 3.9, and the simulations were run on a computer with an Intel(R) Core(TM) i7-7700 CPU @ 3.60 GHz and 16 GB of RAM.

4.1. Identification of Kinetic Model for USV

The minimum trajectory element method proposed in this paper is generated according to the kinetic model of USV. Thus, the real kinetic model of a USV should be obtained before the simulation experiment. In this paper, the SL900 USV independently developed by the Guangzhou Institute of Industrial Intelligence is used to complete the model identification experiment. The USV is equipped with remote control, GPS/IMU integrated navigation, data transmission radio, 4G communication module, microcalculator, and router. It can realize remote control, ground station control, and autonomous navigation. The platform is equipped with two reversible thrusters and belongs to the catamaran category. Its directional control is achieved by the differential rotation of the two propellers. In the control program, we standardized the thrust of the USV to 0 to 1 and the rudder to -0.5 to 0.5. The SL900 USV is shown in Figure 9.

The kinetic model used in this paper is shown in Equation (2), and the unknown parameters in this model need to be obtained through model identification experiments. The simplified method of the USV kinetic model and the related content of the model identification experiment can be found in [40,41,43,44]. Inspired by the literature mentioned above, we designed straight experiment, random acceleration experiment, turning experiment, and zigzag experiment to obtain the sailing data of a USV for identifying the kinetic model of the USV.

Figure 9. The SL900 USV undergoing model identification experiments.

In the straight experiment, the SL900 USV is moving uniformly in a straight line by giving it different fixed thrust. We set the thrust $T = \{0.1, 0.2, \cdots, 1\}$ in the experiment. The acceleration of the USV is zero in the process of uniform linear motion. Thus, we have:

$$a_u u = -b_u T \tag{13}$$

The forward velocity of the USV was collected during the experiment. According to the collected experimental data and the thrust set in advance, the least-square method is used to fit the data of Equation (13) so that the quantitative relationship between a_u and b_u can be obtained. The results of data fitting are shown in Figure 10. We obtain $\hat{b}_u = -2.17666\hat{a}_u$.

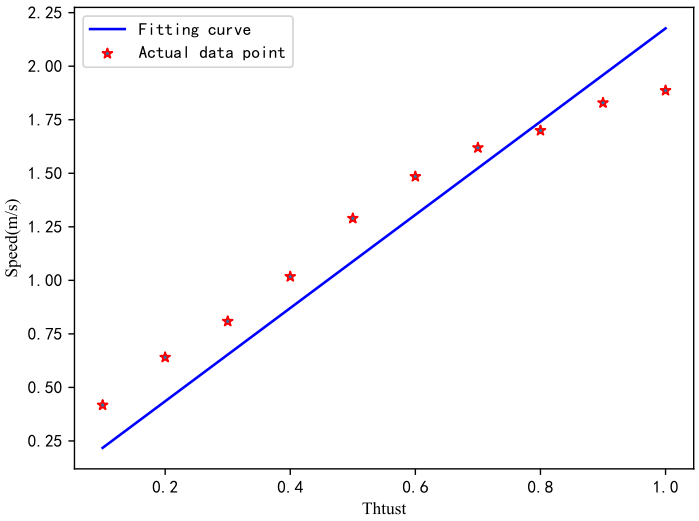

Figure 10. Straight experimental data fitting results ($R^2 = 0.872$).

The random acceleration experiment was to fix the SL900's rudder to zero and randomly set its thrust. This activates its dynamic properties. The forward acceleration, forward velocity, and thrust data of SL900 were recorded during the experiment. Combined with the results of straight experiments, the least-square method can be used to fit the values of a_u and b_u. We obtain $\hat{a}_u = -1.68118$ and $\hat{b}_u = 3.65936$.

In the turning experiment, the thrust of SL900 is fixed at zero, and we set a different rudder for it so that SL900 rotates at uniform yaw velocity. In the experiment, we set the rudder $\delta = \{-0.5, -0.4, \cdots, 0, 0.1, \cdots, 0.5\}$. The yaw acceleration of the SL900 is zero during the cyclotron motion of the uniform yaw velocity. Thus, we have:

$$c_r r = -d_r \delta \tag{14}$$

According to the data of the yaw velocity and the rudder collected during the experiment, the least-square method is used to fit the Equation (14), and the quantitative relationship between c_r and d_r can be obtained. The results of data fitting are shown in Figure 11. We obtain $\hat{d}_r = -1.55183\hat{c}_r$.

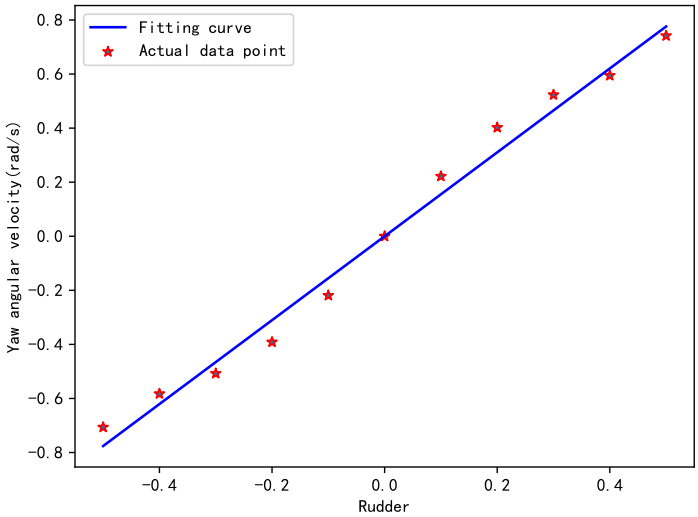

Figure 11. Turning experiment data fitting results ($R^2 = 0.986$).

The zigzag experiment is used to fix the thrust and set the appropriate amount of rudder of the SL900 to make it move in a zigzag shape. Z-type motion can fully stimulate the dynamic characteristics of the USV. Combining the experimental results of the turning experiment and the yaw acceleration, yaw velocity, and rudder during the Z-shaped movement, we can use the least-square method to determine the values of c_r and d_r. We obtain $\hat{c}_r = -3.17724$ and $\hat{d}_r = 4.93053$. The trajectory of SL900 in the zigzag experiment is shown in Figure 12.

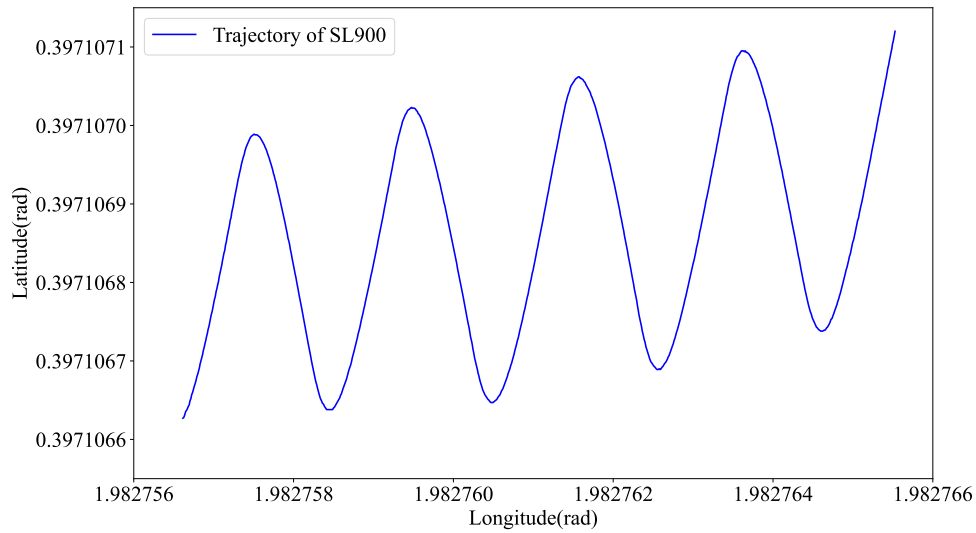

Figure 12. The trajectory of SL900 in the zigzag experiment.

To sum up, the model identification experiment results of the SL900 kinetic model are shown in Table 1. The data are used in the following path planning simulation experiment.

Table 1. Identification results of kinetic model of SL900.

$\hat{a}_u$	$\hat{b}_u$	$\hat{c}_r$	$\hat{d}_r$
−1.68118	3.65936	−3.17724	4.93053

4.2. Path Planning Experiment of a Single USV

The path planning of a single USV is the basis of the formation path planning of USVs. Therefore, we preferentially use a single USV to verify the effectiveness of the algorithm and compare the results of the improved A* algorithm with those of the traditional A* algorithm. The results of the simulation experiment are shown in Figures 13 and 14, where the blue area represents obstacles, the white area is the passable area, the yellow boat symbol is the starting point of path planning, the red boat symbol is the end point of path planning, and the bow of the boat symbol can represent the yaw posture of the USV. The black line is the trajectory planned by the traditional A* algorithm, and the purple line is the trajectory planned by the improved A* algorithm. In addition, the information of yaw on the path planned by the traditional A* algorithm and the improved A* algorithm is shown in Figures 15 and 16. See Tables 2 and 3 for the parameter settings and running results of the simulation experiment.

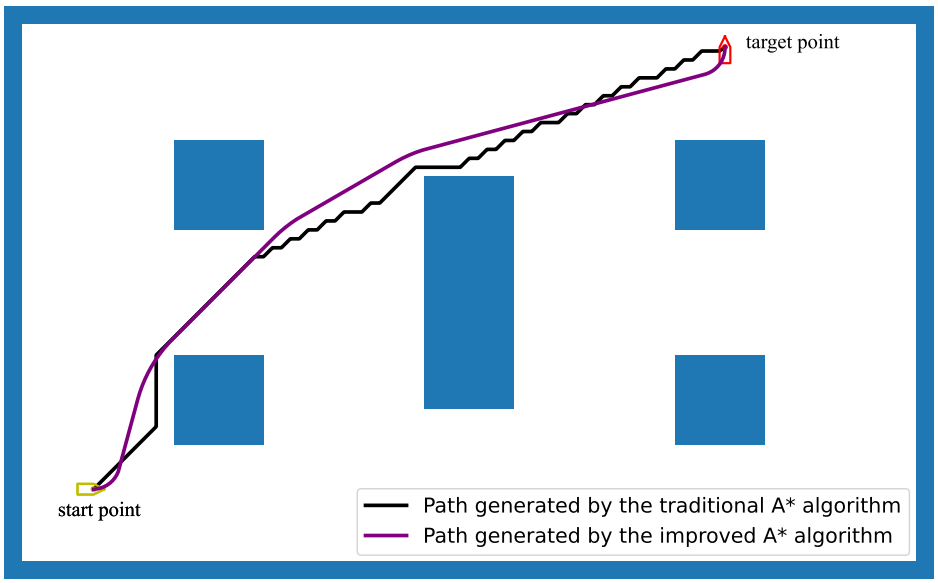

Figure 13. Results of simulation Experiment I of the traditional A* algorithm and the improved A* algorithm.

Table 2. Parameter settings of simulation experiment.

Parameter	Value
The size of grid map	520 m × 320 m
The resolution of the grid	5 m × 5 m
The resolution of yaw	15°
The time interval for generating the minimum trajectory element	4 s
The given thrust	0.5
The discrete quantity of rudder	−0.10, −0.09, ⋯, 0.10

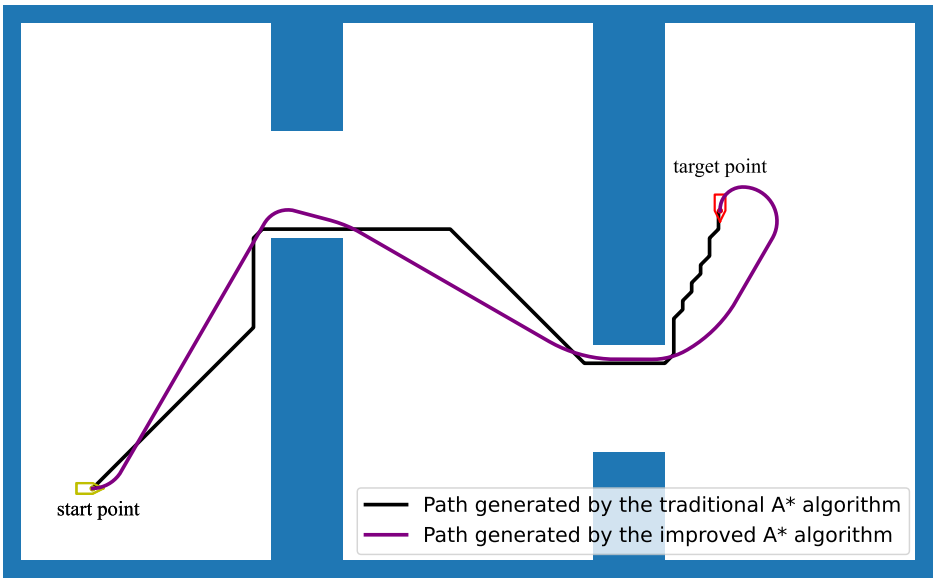

Figure 14. Results of simulation Experiment II of the traditional A* algorithm and the improved A* algorithm.

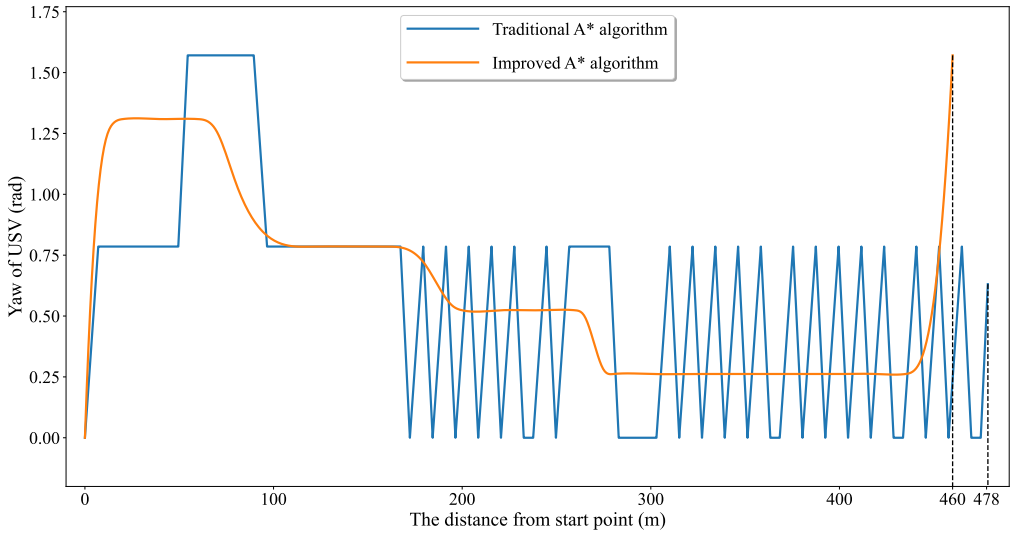

Figure 15. Yaw on the trajectory planned by the traditional A* algorithm and the improved A* algorithm in Experiment I.

From the experimental results, we can find that the path generated by the improved A* algorithm is smoother and has a continuous change in yaw compared with the traditional A* algorithm. The path can be used for USV tracking without subsequent processing. In addition, the improved A* algorithm takes into account the yaw of the starting and ending points of USV and is applicable to more scenarios (such as berths). However, the improved A* algorithm searches more state points and spends more time than the traditional A* algorithm because of the yaw constraint. From Table 3, we can find that the improved A* algorithm by querying the heuristic value map can reduce searching a large number of state points and save the searching time. It is acceptable to spend a little more time in the path search because the paths conform to the kinetic constraints of the USV.

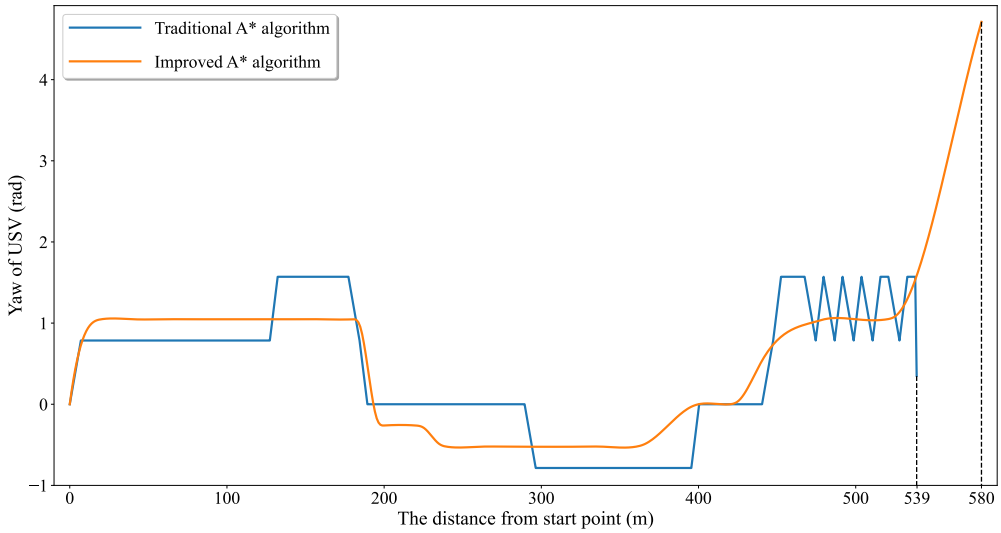

Figure 16. Yaw on the trajectory planned by the traditional A* algorithm and the improved A* algorithm in Experiment II.

Table 3. The average time of each run of the algorithm over 1000 times.

No. of Experiment	Algorithm	Length of Path	Points of Search
I	Traditional A* algorithm	478	1209
I	Improved A* algorithm with heuristic value map	460	7052
I	Improved A* algorithm without heuristic value map	460	21,673
II	Traditional A* algorithm	539	2402
II	Improved A* algorithm with heuristic value map	580	18,583
II	Improved A* algorithm without heuristic value map	580	80,722

4.3. Path Planning Experiment of Unmanned Surface Vehicles Formation

In order to verify the effectiveness of the formation path planning algorithm, we design four simulation experiments of formation navigation path planning, formation reconfiguration path planning, path planning for formation through narrow passageway, and formation whole flow path planning. The experimental results are shown in Figures 17–20. Black, green, and yellow USVs were set up in the formation of the above four experiments. The trajectories of black, green, and yellow USVs planned by the algorithm are represented by lines of corresponding colors in the figure. In addition, the blue area in the figure is the impassable area, and the white area is the passable area. The relevant parameter settings of this experiment are the same as the path planning of a single USV, as shown in Table 2.

As shown in Figure 17, we set the USV formation as a one-line in the formation navigation path planning experiment, and use Algorithm 4 to plan the trajectory of each USV from the starting position of the formation to the target position. As shown in Figure 18, we set the initial formation of USV as a one-line and the target formation as a triangle in the path planning experiment of formation reconfiguration, and use Algorithm 5 to plan the trajectory of each USV from the initial position to the target position. Figure 19 shows the path planning results of USV formation through the narrow passage. Figure 21 illustrates

the relationship between the distance between any two USVs in this process as a function of time. The initial state A of USV formation is a one-line formation and maintains the one-line formation to navigate to point B. The USV formation encounters an impassable narrow passage ahead, so it changes its formation to a line formation at point C. The USV formation passes through the narrow passage to point D in a line formation, and finally changes its formation back to a one-line formation. The path planning from A to B and C to D uses Algorithm 4, and the path planning from B to C and D to E uses Algorithm 5. As shown in Figure 20, the whole process path planning experiment of formation refers to the path planning of a complete process of formation generation, formation navigation, and formation reconfiguration. The distance between each two USVs in formation is shown in Figure 22. In the beginning, the position of the three USVs was random and there was no fixed formation. They are marked with the letter A in the picture. After receiving the task from the mission module, the USV began to use Algorithm 5 to plan the trajectory to target point B to form a triangular formation, and then used Algorithm 4 to plan the trajectory to sail to point C in a triangular formation. Finally, Algorithm 5 was used to plan the trajectory from point C to point D to complete the transformation from triangular formation to line formation. As can be seen from Figure 22, when USVs sail in formation, the distance between each two USVs remains stable, that is, a stable formation can be formed between USVs. In addition, the minimum distance between USVs is 4.49 m, which indicates that the algorithm can complete collision avoidance between USVs.

It can be seen from the simulation results that the path planned by the algorithm is smooth and continuous, conforms to the kinetic constraints of USVs, and can safely avoid known obstacles and other USVs in formation. This method can provide a globally safe path with kinetic constraints for USV formation navigation and formation reconstruction.

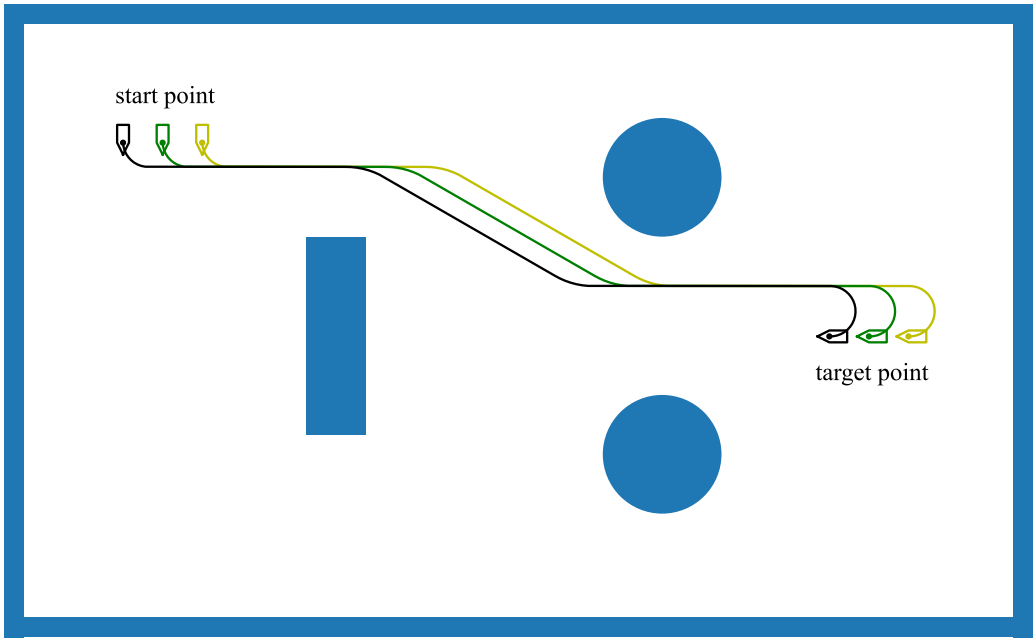

Figure 17. Path planning of USV formation sailing in line formation.

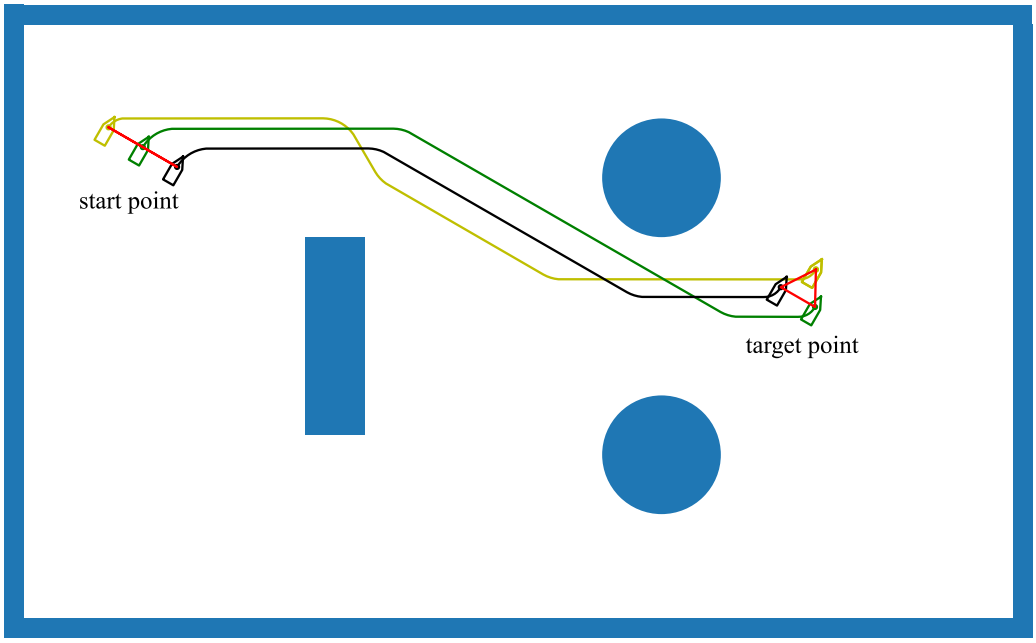

Figure 18. The path planning of USV formation from one formation to triangle formation.

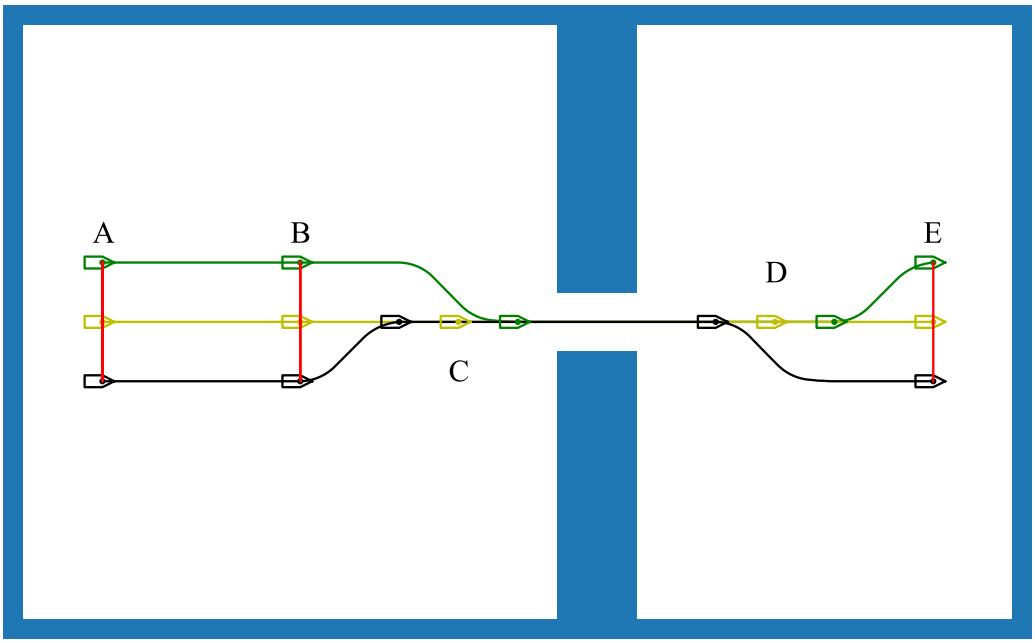

Figure 19. The path planning for USV formation through narrow passageway.

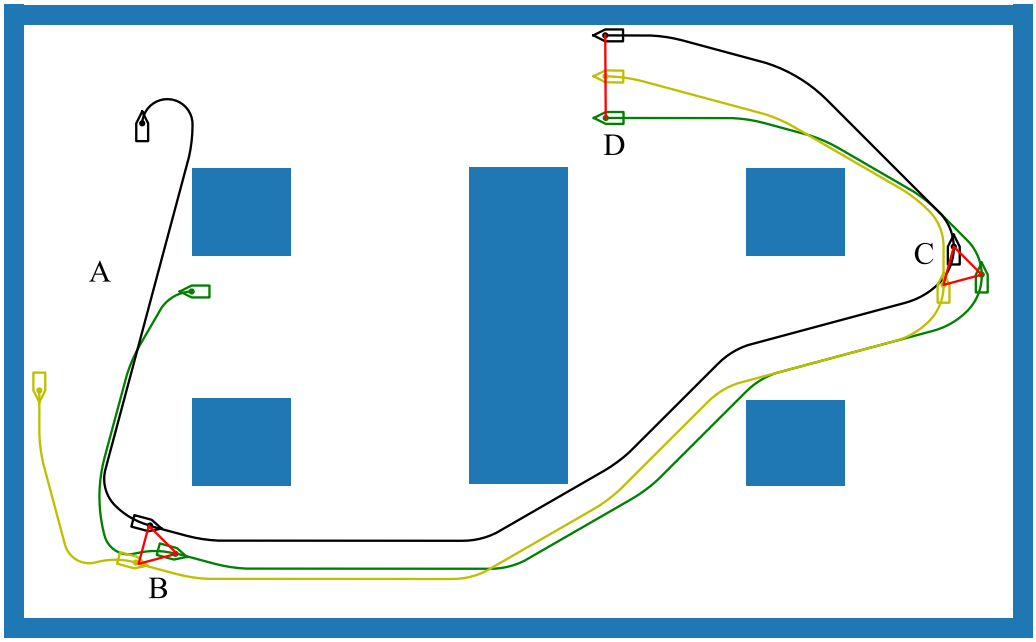

Figure 20. Simulation of whole process path planning of USV formation.

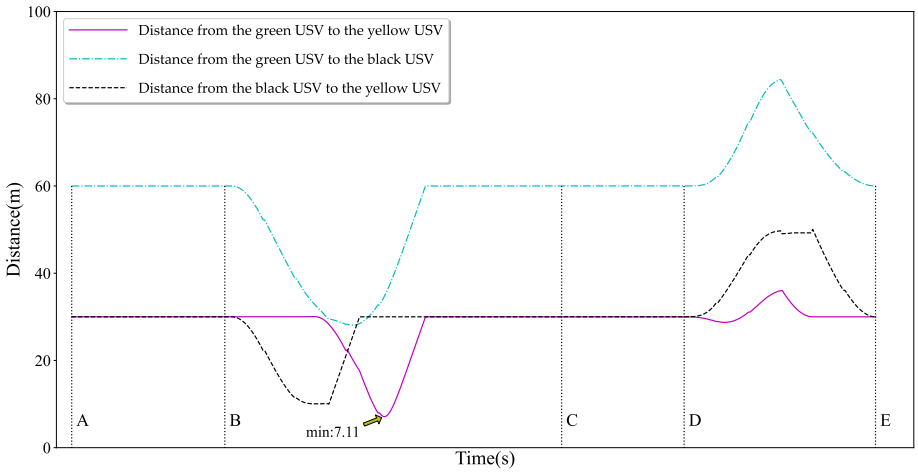

Figure 21. Distance between each two USVs in path planning for passing narrow passageway.

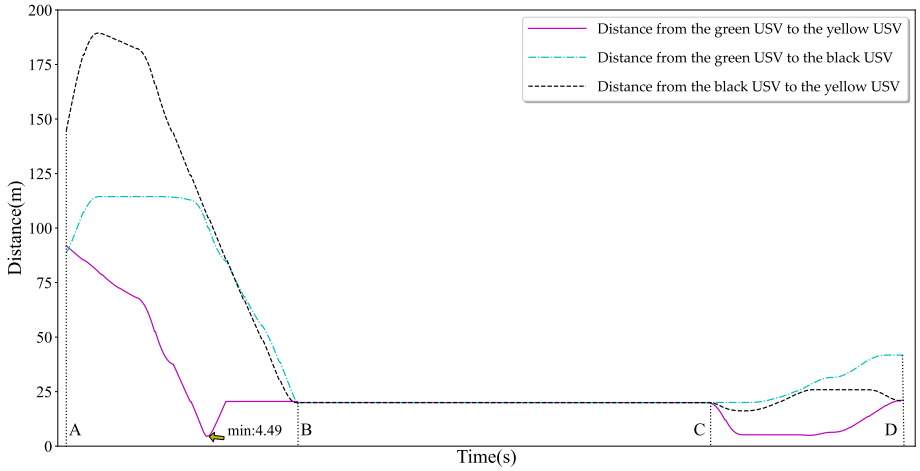

Figure 22. Distance between each two USVs in the whole process of path planning.

5. Discussion

In this section, we discuss the advantages and disadvantages of the proposed method.

The method has distinct advantages. First, since we take into account the additional yaw constraint and use the dynamics equations of the USV to generate the minimum trajectory elements, the path output by the proposed algorithm conforms to the kinetic constraints of USV. We believe that this is the main contribution of this paper because it allows USVs to track directly without subsequent processing of the path. In addition, we divide the USV formation path planning into formation navigation and formation reconfiguration path planning for the first time, and we give solutions for these two kinds of path planning, i.e., the method proposed in this paper can solve both types of problems.

The algorithm also has some inherent limitations. First, the improved A* algorithm requires more computational resources compared to the traditional A* algorithm. This is understood because we extend the state space from two dimensions to three dimensions. We propose the method of building heuristic value maps to speed up the path search, which leads to some improvement of the problem. In addition, we only apply rudder in the first half of the minimum trajectory element to ensure that the yaw and velocity are continuous on the path stitched by the minimum trajectory elements when generating the minimum trajectory elements. This makes the rudder change frequently on the whole path, which is not friendly to the actuator. Finally, the robustness of the algorithm needs to be improved. Since we adopt a formation-wide obstacle avoidance scheme, the algorithm needs to replan the path for each USV when only one or some USVs encounter obstacles outside the global information.

6. Conclusions and Future Works

For the formation path planning problem considering USV kinetic constraints, an improved A* algorithm is proposed in this paper. The state space of the USV is extended to three dimensions by considering the yaw constraint of the USV on the basis of the two-dimensional spatial position of the plane. The state space model of the USV is introduced by using the kinetic model of the USV. The minimum trajectory elements of the USV are generated from the state space model. The optimal path of the USV is searched using the A* algorithm, which consists of the minimum trajectory elements conforming to the kinetic constraints of the USV. The heuristic value map considering the obstacles and the beginning and end yaw is established before the search to obtain the heuristic value closer to the actual cost. The search speed can be accelerated by querying the heuristic value map to obtain the heuristic value. The improved A* algorithm generates a smoother path with a continuous yaw compared with the traditional A* algorithm, which enables the USV to track directly without subsequent processing.

We divide formation path planning into formation navigation and formation reconfiguration. The formation structure of the USV formation uses the virtual structure method. For formation navigation path planning, the path of the virtual rigid body coordinate system is planned considering the formation of the USV, and the path of each USV is generated according to the position of each USV under the virtual rigid body coordinate system. During formation reconfiguration path planning, the path of each USV is planned in turn, and each minimum trajectory element in the path is added to the obstacle map at the corresponding moment after each USV has planned its path to achieve collision avoidance among USVs.

In addition, we conducted model identification experiments for SL900 USV and simulation experiments based on the model to verify the effectiveness of the algorithm. Experimental results show that this approach can provide a globally safe path with kinetic constraints for USV formation navigation and formation reconstruction.

Finally, we discussed the advantages and disadvantages of the proposed method.

For future work, the algorithm proposed in this paper will be implemented on an actual USV platform. The corresponding task allocation module and task execution module described in Figure 1 will be designed. After the construction of the USV platform is

completed, sea trials will be performed. The algorithm will be further modified according to the results of the sea trials.

Author Contributions: Conceptualization, T.S. and J.X. (Jinchao Xiao); methodology, T.S.; software, T.S.; validation, T.S.; formal analysis, J.X. (Jinchao Xiao); investigation, J.X. (Junfeng Xiong); resources, H.X.; data curation, H.X.; writing—original draft preparation, Z.W.; writing—review and editing, Z.W.; visualization, H.X.; supervision, J.X. (Jinchao Xiao) and J.X. (Junfeng Xiong); project administration, J.X. (Jinchao Xiao) and J.X. (Junfeng Xiong); funding acquisition, J.X. (Junfeng Xiong). All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Guangdong Basic and Applied Basic Research Foundation (grant number: 2020A1515010584), Key-Area Research and Development Program of Guangdong Province (grant number: 2020B1111010002) and Nansha District Science and Technology Project.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data that support the findings of this study are available from the corresponding author upon reasonable request.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Yuh, J.; Marani, G.; Blidberg, D.R. Applications of marine robotic vehicles. *Intell. Serv. Robot.* **2011**, *4*, 221. [CrossRef]
2. Liu, Z.; Zhang, Y.; Yu, X.; Yuan, C. Unmanned surface vehicles: An overview of developments and challenges. *Annu. Rev. Control* **2016**, *41*, 71–93. [CrossRef]
3. Campbell, S.; Naeem, W.; Irwin, G.W. A review on improving the autonomy of unmanned surface vehicles through intelligent collision avoidance manoeuvres. *Annu. Rev. Control* **2012**, *36*, 267–283. [CrossRef]
4. Felski, A.; Zwolak, K. The Ocean-Going Autonomous Ship—Challenges and Threats. *J. Mar. Sci. Eng.* **2020**, *8*, 41. [CrossRef]
5. Savitz, S.; Blickstein, I.; Buryk, P.; Button, R.W.; DeLuca, P.; Dryden, J.; Mastbaum, J.; Osburg, J.; Padilla, P.; Potter, A. *US Navy Employment Options for Unmanned Surface Vehicles (USVs)*; Technical Report; Rand National Defense Research Inst.: Santa Monica, CA, USA, 2013.
6. Liu, Y.; Bucknall, R. Path planning algorithm for unmanned surface vehicle formations in a practical maritime environment. *Ocean Eng.* **2015**, *97*, 126–144. [CrossRef]
7. Xue, Y.; Sun, J.Q. Solving the path planning problem in mobile robotics with the multi-objective evolutionary algorithm. *Appl. Sci.* **2018**, *8*, 1425. [CrossRef]
8. Deb, K. Multi-objective optimization. In *Search Methodologies*; Springer: Berlin/Heidelberg, Germany, 2014; pp. 403–449.
9. Kumar, R.; Singh, S.; Bilga, P.S.; Singh, J.; Singh, S.; Scutaru, M.L.; Pruncu, C.I. Revealing the benefits of entropy weights method for multi-objective optimization in machining operations: A critical review. *J. Mater. Res. Technol.* **2021**, *10*, 1471–1492. [CrossRef]
10. Pirouz, B.; Khorram, E. A computational approach based on the ε-constraint method in multi-objective optimization problems. *Adv. Appl. Stat.* **2016**, *49*, 453. [CrossRef]
11. Uskov, A.; Serdyukova, N.A.; Serdyukov, V.I.; Heinemann, C.; Byerly, A. Multi objective optimization of VPN design by linear programming with risks models. *Int. J. Knowl.-Based Intell. Eng. Syst.* **2016**, *20*, 175–188. [CrossRef]
12. Konak, A.; Coit, D.W.; Smith, A.E. Multi-objective optimization using genetic algorithms: A tutorial. *Reliab. Eng. Syst. Saf.* **2006**, *91*, 992–1007. [CrossRef]
13. Nguyen, T.V.; Huynh, N.T.; Vu, N.C.; Kieu, V.N.; Huang, S.C. Optimizing compliant gripper mechanism design by employing an effective bi-algorithm: Fuzzy logic and ANFIS. *Microsyst. Technol.* **2021**, *27*, 3389–3412. [CrossRef]
14. Wang, C.N.; Yang, F.C.; Nguyen, V.T.T.; Nguyen, Q.M.; Huynh, N.T.; Huynh, T.T. Optimal Design for Compliant Mechanism Flexure Hinges: Bridge-Type. *Micromachines* **2021**, *12*, 1304. [CrossRef] [PubMed]
15. Qu, D.K.; Du, Z.J.; Xu, D.G.; Xu, F. Research on path planning for a mobile robot. *Robot* **2008**, *30*, 97–101.
16. Zuo, L.; Yan, W.; Cui, R.; Gao, J. A coverage algorithm for multiple autonomous surface vehicles in flowing environments. *Int. J. Control Autom. Syst.* **2016**, *14*, 540–548. [CrossRef]
17. Qiu, H.; Duan, H. Receding horizon control for multiple UAV formation flight based on modified brain storm optimization. *Nonlinear Dyn.* **2014**, *78*, 1973–1988. [CrossRef]
18. Vicmudo, M.P.; Dadios, E.P.; Vicerra, R.R.P. Path planning of underwater swarm robots using genetic algorithm. In Proceedings of the 2014 International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM), Palawan, Philippines, 12–16 November 2014; pp. 1–5.
19. Tam, C.; Bucknall, R. Cooperative path planning algorithm for marine surface vessels. *Ocean Eng.* **2013**, *57*, 25–33. [CrossRef]

20. Sang, H.; You, Y.; Sun, X.; Zhou, Y.; Liu, F. The hybrid path planning algorithm based on improved A* and artificial potential field for unmanned surface vehicle formations. *Ocean Eng.* **2021**, *223*, 108709. [CrossRef]
21. Ouyang, Z.; Wang, H.; Huang, Y.; Yang, K.; Yi, H. Path planning technologies for USV formation based on improved RRT. *Chin. J. Ship Res.* **2020**, *15*, 18–24.
22. Wang, Yuewu. Research on the Technologies of Path Planning for USV Formation on Fast Marching Method. Ph.D. Thesis, Harbin Engineering University: Harbin, China, 2015.
23. Hao, Y.; Agrawal, S.K. Planning and control of UGV formations in a dynamic environment: A practical framework with experiments. *Robot. Auton. Syst.* **2005**, *51*, 101–110. [CrossRef]
24. Duan, H.B.; Ma, G.J.; Luo, D.L. Optimal formation reconfiguration control of multiple UCAVs using improved particle swarm optimization. *J. Bionic Eng.* **2008**, *5*, 340–347. [CrossRef]
25. Qu, H.; Xing, K.; Alexander, T. An improved genetic algorithm with co-evolutionary strategy for global path planning of multiple mobile robots. *Neurocomputing* **2013**, *120*, 509–517. [CrossRef]
26. Asl, A.N.; Menhaj, M.B.; Sajedin, A. Control of leader–follower formation and path planning of mobile robots using Asexual Reproduction Optimization (ARO). *Appl. Soft Comput.* **2014**, *14*, 563–576.
27. Zhou, C.; Gu, S.; Wen, Y.; Du, Z.; Xiao, C.; Huang, L.; Zhu, M. The review unmanned surface vehicle path planning: Based on multi-modality constraint. *Ocean Eng.* **2020**, *200*, 107043. [CrossRef]
28. Song, R.; Liu, Y.; Bucknall, R. A multi-layered fast marching method for unmanned surface vehicle path planning in a time-variant maritime environment. *Ocean Eng.* **2017**, *129*, 301–317. [CrossRef]
29. Yuxuan, Z.; Lei, G.; Xin, Z.; Yan, P.; Yi, Y.; Xiaomao, L. Complete coverage path planning of USV used for mapping round island. *J. Shanghai Univ.* **2017**, *23*, 17–26.
30. Lazarowska, A. A discrete artificial potential field for ship trajectory planning. *J. Navig.* **2020**, *73*, 233–251. [CrossRef]
31. Naeem, W.; Henrique, S.C.; Hu, L. A reactive COLREGs-compliant navigation strategy for autonomous maritime navigation. *IFAC-PapersOnLine* **2016**, *49*, 207–213. [CrossRef]
32. Hart, P.E.; Nilsson, N.J.; Raphael, B. A formal basis for the heuristic determination of minimum cost paths. *IEEE Trans. Syst. Sci. Cybern.* **1968**, *4*, 100–107. [CrossRef]
33. Dijkstra, E.W. A note on two problems in connexion with graphs. *Numer. Math.* **1959**, *1*, 269–271. [CrossRef]
34. Dechter, R.; Pearl, J. Generalized best-first search strategies and the optimality of A. *J. ACM (JACM)* **1985**, *32*, 505–536. [CrossRef]
35. Pearl, J. *Intelligent Search Strategies for Computer Problem Solving*; Addison-Wesley Pub. Co.: Boston, MA, USA 1984.
36. Fan, X.; Singh, S.; Oppolzer, F.; Nettleton, E.; Hennessy, R.; Lowe, A.; Durrant-Whyte, H. Integrated planning and control of large tracked vehicles in open terrain. In Proceedings of the 2010 IEEE International Conference on Robotics and Automation, Anchorage, AK, USA, 3–7 May 2010; pp. 4424–4430.
37. Fraichard, T.; Scheuer, A. From Reeds and Shepp's to continuous-curvature paths. *IEEE Trans. Robot.* **2004**, *20*, 1025–1035. [CrossRef]
38. Knepper, R.A.; Kelly, A. High Performance State Lattice Planning Using Heuristic Look-Up Tables. In Proceedings of the IROS, Hong Kong, China, 5–7 January 2016; pp. 3375–3380.
39. Du, Z.; Wen, Y.; Xiao, C.; Huang, L.; Zhou, C.; Zhang, F. Trajectory-cell based method for the unmanned surface vehicle motion planning. *Appl. Ocean Res.* **2019**, *86*, 207–221. [CrossRef]
40. Sonnenburg, C.R.; Woolsey, C.A. Modeling, identification, and control of an unmanned surface vehicle. *J. Field Robot.* **2013**, *30*, 371–398. [CrossRef]
41. Zhang, M.; He, Y.; Xiong, J. Research on the Unmanned Surface Vehicle Kinetics Model for Automatic Berthing. In *Advances in Guidance, Navigation and Control*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 3659–3670.
42. Li, Z.; Xian, B. Robust distributed formation control of multiple unmanned aerial vehicles based on virtual structure. *Control Theory Appl.* **2020**, *37*, 2423–2431.
43. Mu, D.; Wang, G.; Fan, Y.; Sun, X.; Qiu, B. Modeling and Identification for Vector Propulsion of an Unmanned Surface Vehicle: Three Degrees of Freedom Model and Response Model. *Sensors* **2018**, *18*, 1889. [CrossRef] [PubMed]
44. Han, J.; Xiong, J.; He, Y.; Gu, F.; Li, D. Nonlinear Modeling for a Water-Jet Propulsion USV: An Experimental Study. *IEEE Trans. Ind. Electron.* **2017**, *64*, 3348–3358. [CrossRef]