2023 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)
June 28-30, 2023. Seattle, Washington, USA

Efficient Trajectory Planning and Control for USV with Vessel Dynamics and Differential Flatness

Tao Huang[1,2], Zhenfeng Xue[1,2], Zhe Chen[1,2], Yong Liu[1,2]

***Abstract*— Unmanned surface vessels (USVs) are widely used in ocean exploration and environmental protection. To ensure that USV can successfully perform its mission, trajectory planning and motion tracking are the two most critical technologies. This paper proposes a novel trajectory generation and tracking method for USV based on optimization theory. Specifically, the USV dynamic model is combined with differential flatness, so that the trajectory can be generated by dynamic RRT* in a linear invariant system expression form under the objective of optimal boundary value. We adjust the trajectory through local optimization to reduce the number of samples and improve efficiency. The dynamic constraints are considered in the optimization process so that the generated trajectory conforms to the kinematic characteristics of the under-actuated hull, making tracking easier. Finally, motion tracking is added with model predictive control under a sequential quadratic programming problem. Simulated results show that the planned trajectory is more consistent with the kinematic characteristics of USV, and the tracking accuracy remains at a higher level.**

I. Introduction

Unmanned surface vehicles (USVs) are designed to relieve human labor in various water surface missions, such as surface cleaning, cyanobacteria treatment, cargo transportation, and military reconnaissance. It mainly perceives and maps the surrounding environment through shipborne radar or camera and then automatically performs path planning and motion control [1], [2], [3]. In the traveling process, the USV must accurately avoid various static and dynamic environmental obstacles. To ensure that USV can complete its mission autonomously successfully, obstacle avoidance trajectory planning [4] and motion control [5] are the two most critical technologies.

However, current trajectory planning algorithms are usually modified directly from other similar unmanned platforms, such as unmanned aerial vehicles (UAV) or automated guided vehicles (AGV), which ignores the inherent kinematic characteristics of USV and thus degrades the motion tracking control performance. We usually regard the USV as a system with a three degree of freedom, *i.e.*, surge, sway, and yaw, but the drive dimension only includes propeller thrust along the surge axis and its torque along the yaw axis. This makes the generated path hard to be tracked for USV due to the lack of relevance between path planning and tracking.

This paper proposes a novel trajectory planning and control method designed explicitly for USV. Under the circumstance that the global map is assumed to be known, the dynamic RRT* algorithm is first applied to generate the initial obstacle avoidance trajectory. In the process of constructing the sampling interval, we propose to combine the Dubins curve [6] and A* algorithm because the smoothness of the Dubins curve fit the hull dynamics well. After that, the path planning problem is expressed as a linear time invariant (LTI) form, and trajectory optimization is established concerning the path's spatial quality, such as smoothness, obstacle avoidance, and vessel dynamics. Different from the existing methods [4], [7] that directly use the dynamic model of the USV system as the equality constraint in the trajectory generation process, which causes a vast computational burden and unstable generation results. Inspired by related work on UAV differential flatness [8], [9], we use differential flatness to describe the USV dynamic system to solve the optimization problem. Our proposed method dramatically improves the optimization efficiency, and the usage of differential flatness can connect path planning with motion control well and make the subsequent motion tracking control more accurate.

The effect of the proposed path planning algorithm is illustrated in Fig. 1, from which the USV can travel across the map of multiple obstacles efficiently. The generated path is smooth enough to be consistent with the motion characteristics of the USV. In the subsequent motion control process, we successfully combine the differential flatness of the system with the nonlinear model predictive control (NMPC). A quadratic optimization problem about the error between a given trajectory state and the forward prediction state is constructed and cyclically solved. The tracking accuracy remains at a high level during the rapid operation of the USV, no matter whether there is directional water interference.

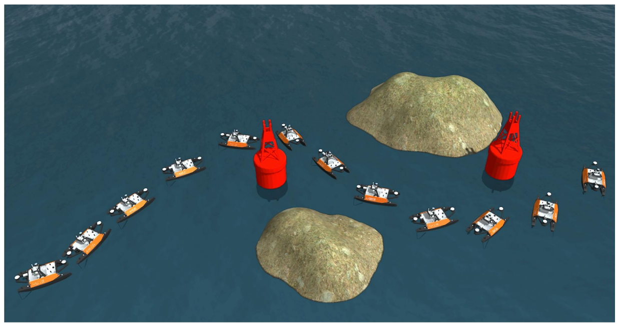
Fig. 1. A composite image of USV sailing in obstacle environment with planned trajectory that fits the dynamic characteristics.

[1]Tao Huang, Zhenfeng Xue, Zhe Chen and Yong Liu are with Institute of Cyber-Systems and Control, Zhejiang University, Hangzhou, China
[2]Tao Huang, Zhenfeng Xue, Zhe Chen and Yong Liu are Intelligent Perception and Control Center, Huzhou Institute of Zhejiang University, Huzhou, China (Zhenfeng Xue and Yong Liu are the corresponding authors, zfxue0903@zju.edu.cn, yongliu@iipc.zju.edu.cn)

2023 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM) | 978-1-6654-7633-1/23/$31.00 ©2023 IEEE | DOI: 10.1109/AIM46323.2023.10196111

II. Related works

A. Path planning for USV

The motion planning algorithm for USV includes two essential contents, *i.e.*, path generation, and obstacle avoidance. Path generation makes the USV quickly acquire the feasible route to the mission target based on the map information. Han *et al.* [10] proposed Non-uniform Theta* to establish the mapping relation containing the parent node and minimum path-cost of each cell. In addition to the traditional graph search methods [11], there are also some valuable applications based on reinforcement learning [12] in the path planning problems. Wen *et al.* [4] proposed a hierarchical trajectory planning that constructs an optimization problem based on RRT* algorithm and dynamic model of USV. Du *et al.* [7] modeled the problem in the state space and proposed to sample a good trajectory based on motion primitives. In order to effectively describe obstacle information, various improved artificial potential field (APF) methods [13] are used in USV obstacle description, and obstacle avoidance strategy in a complex water environment. Compared with the above methods, this paper proposes a USV trajectory optimization algorithm that combines the Dubins curve [6] and the dynamic RRT* to improve the sampling efficiency. While stably generating the navigation trajectory that meets the motion characteristics of USV, it ensures real-time performance and quick response to unknown obstacles.

B. Motion control for USV

After the feasible navigation trajectory is obtained, robust trajectory tracking is another key component to realize the autonomous navigation of USV. In the field of USV motion control, different control methods have been proposed, such as sliding mode control based on adaptive neural network(NN) [14], deep reinforcement learning-based adaptive control [15] and model predictive control [3]. Sarda *et al.* [16] design a nonlinear back stepping PD controller to reach and maintain a specific configuration of heading and position in environments of uncertain wind, current and wave disturbances. These control methods deal well with the problem of robust tracking of the USV path in the presence of external disturbances. However, these works generally only focus on the tracking of the USV trajectory or simply consider the planning task of the USV path, and ignore the efficiency of directly using the generated path for tracking control. In Han's work [17], the kinematic model of USV is added to path generation to make the navigation more suitable for USV tracking control. However, the computational burden and algorithm stability become challenging problems. Referring to [8], [18], this paper proposes the differential flatness property of the USV motion system to simplify the kinematic model as an algebraic expression of the position and its higher order derivatives. It effectively connects the navigation motion planning with the USV tracking control problem so that the USV can more stably follow the trajectory and improve the stability and success rate.

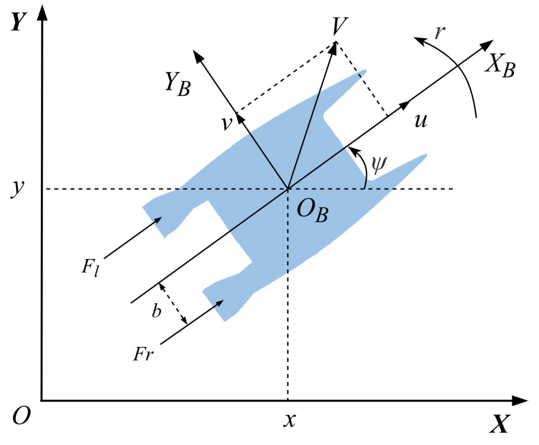

Fig. 2. USV earth-fixed and body-fixed coordinate frames.

III. Description of USV system

A USV is an under-actuated system with six degrees of freedom. When its dynamic constraints are directly embedded into path planning and motion control, the results could be unstable. One feasible solution is to use flatness property to describe the system.

A. USV discrete model

As shown in Fig. 2, a USV with kinematics and dynamics in earth-fixed and body-fixed frames can be modeled by

$$\begin{cases} \dot{\boldsymbol{\eta}} = \boldsymbol{R}(\boldsymbol{\eta})\boldsymbol{\nu} \\ \boldsymbol{M}\dot{\boldsymbol{\nu}} = \boldsymbol{\tau} - \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} - \boldsymbol{D}\boldsymbol{\nu}, \end{cases} \tag{1}$$

where $\boldsymbol{\eta} = [x, y, \varphi]^\top$ and $\boldsymbol{\nu} = [u, v, r]^\top$ are position, heading and velocity vectors in the earth-fixed and body-fixed frames respectively. $\boldsymbol{R}(\boldsymbol{\eta})$ is the rotation matrix, and $\boldsymbol{M}$ is positive-definite inertia matrix. $\boldsymbol{C}(\boldsymbol{\nu})$ is the Coriolis and centripetal force matrix, and $\boldsymbol{D}$ is linear hydrodynamic damping matrix. $\boldsymbol{\tau} = [\tau_u, \tau_v, \tau_r]^\top$ is the thrust matrix.

For simplicity, considering a catamaran, we suppose the hull is symmetrical about the u-axis and v-axis. The constraint of v-axis symmetry is over strict, but the practice has proved it is almost feasible for a catamaran. Thus, the relevant matrix can be expressed as

$$\boldsymbol{M} = diag\{m_1, m_1, m_2\}, \tag{2}$$

$$\boldsymbol{C}(\boldsymbol{\nu}) = \begin{bmatrix} 0 & 0 & -m_2 v \\ 0 & 0 & m_1 u \\ m_2 v & -m_1 u & 0 \end{bmatrix}, \tag{3}$$

$$\boldsymbol{D} = diag\{d_1, d_2, d_3\}, \tag{4}$$

where $m_1 = m + m_a$, $m_2 = I_z + n_z$. m_a and n_z represent the added mass due to water flow. The thrust matrix can be expressed as

$$\boldsymbol{\tau} = \begin{bmatrix} F_r + F_l \\ 0 \\ b \cdot (F_r - F_l) \end{bmatrix}. \tag{5}$$

Combined with the above equations, the USV dynamic model is rewritten as the form of state equation as follow

$$\dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t), \boldsymbol{u}(t)), \tag{6}$$

where the state and input of system is $\boldsymbol{x} = [x, y, \varphi, u, v, r]^\top$ and $\boldsymbol{u} = [\tau_u, \tau_r]^\top$ respectively. In order to reflect the motion characteristics of USV in discrete time domain, we use the explicit 4th order Runge Kutta method to calculate the form

$$\boldsymbol{x}_{i+1}(t) = f_{RK4}(\boldsymbol{x}_i(t), \boldsymbol{u}_i(t), \Delta t), \tag{7}$$

where Δt is the discrete step. There are unknown hydrodynamic parameters $\boldsymbol{h} = [m_1, m_2, d_1, d_2, d_3]^\top$ in the above USV system model, which will be used in subsequent planning and control methods. Based on Eq. 7, we use sampled USV system motion state data $\boldsymbol{X}_s = [\boldsymbol{x}_0, \boldsymbol{x}_1, ..., \boldsymbol{x}_k]^\top$ and thrust data $\boldsymbol{U}_s = [\boldsymbol{u}_0, \boldsymbol{u}_1, ..., \boldsymbol{u}_k]^\top$with discrete step Δt to estimate $\boldsymbol{h}$ with the following optimization problem

$$\begin{aligned} \min_{\boldsymbol{h}} \sum_{i=0}^{k} & \boldsymbol{\varepsilon}(i)^\top \boldsymbol{\Omega} \boldsymbol{\varepsilon}(i) \\ s.t. \quad & \boldsymbol{h}_l \le \boldsymbol{h} \le \boldsymbol{h}_u \\ \boldsymbol{x}_{i+1}^{pred} = f_{RK4}(&\boldsymbol{X}_s(i), \boldsymbol{U}_s(i), \Delta t, \boldsymbol{h}), i \in [0, k-1] \\ \boldsymbol{\varepsilon}(i) = &\boldsymbol{X}_s(i) - \boldsymbol{x}_i^{pred}, \end{aligned} \tag{8}$$

where $\boldsymbol{\varepsilon}(i)$ is the deviation between real state data $\boldsymbol{X}_s(i)$ and predicted data $\boldsymbol{x}_i^{pred}$, $\boldsymbol{h}_l$ and $\boldsymbol{h}_u$ are the lower and upper boundary of hydrodynamic parameters respectively and $\boldsymbol{\Omega}$ is weight matrix. The solution can be solved by Gauss Newton Hessian approximation method with the help of Casadi [19] optimization library and be used in section VI finally.

B. Differential flatness

Due to the USV system is under-actuated and has nonholonomic constraints, real-time planning and accurate trajectory control still exist challenges. Wen *et al.* [4] try to insert the dynamic model of the USV system as the equality constraint into trajectory generation so that the generated trajectory meets the motion constraints of USV. However, this leads to a heavy computational burden and unstable results. Similar to [20], it is feasible to simplify the problem that USV model constraints need to be considered in trajectory generation by using the flat property of the system.

According to [5], USV system is controllable and has differential flatness. Then the system has a flat output $\boldsymbol{p}(t) = [x(t), y(t)]^\top \in \mathcal{R}^m$, implying the state and input can be expressed in the algebraic form of derivatives using only $\boldsymbol{p}(t)$ and its finite order [18],

$$\begin{cases} \boldsymbol{x}(t) = \phi_0(\boldsymbol{p}(t), \dot{\boldsymbol{p}}(t), \ddot{\boldsymbol{p}}(t), \boldsymbol{p}^{(3)}(t)), \\ \boldsymbol{u}(t) = \phi_1(\boldsymbol{p}(t), \dot{\boldsymbol{p}}(t), \ddot{\boldsymbol{p}}(t), \boldsymbol{p}^{(3)}(t), \boldsymbol{p}^{(4)}(t)), \end{cases} \tag{9}$$

where ϕ_0 and $\phi_1$are the flatness transformation. The highest derivative order of $\boldsymbol{p}$ in Eq. 9 is equal to 4. In other words, the polynomial expression for $\boldsymbol{p}$ needs to reach at least 5-th order to ensure smoothness [8].

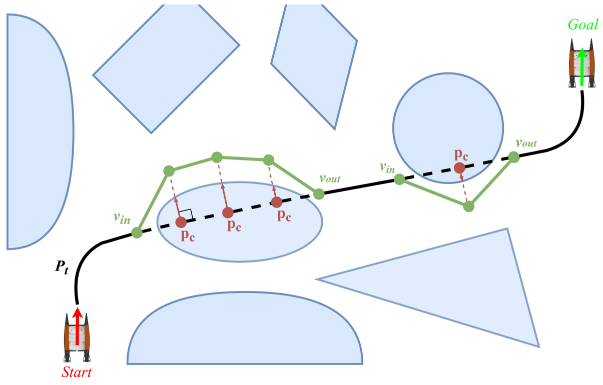

Fig. 3. An illustration of the path search process. The black path obtained by Dubins curve has several collision points p_c, which will be moved outside the collision with a safe distance, resulting the green path.

Algorithm 1 $KinodynamicTrajectoryPlan$

```
Input: Environment $env$, SamplingTree $\zeta$, PlanningState $\mathcal{X}$
Output: Trajectory $\varGamma$
1:  Initialize: $\zeta_s \leftarrow \mathcal{X}_{start}, \zeta_g \leftarrow \mathcal{X}_{goal}$
2:  $\mathcal{P}_t \leftarrow \boldsymbol{TopoPathFind}(env)$
3:  for i = 1 to n do
4:      $\mathcal{X}_{rand} \leftarrow \boldsymbol{SampleState}(env, \boldsymbol{p}_t)$
5:      $\mathcal{X}_{front}, \mathcal{X}_{back} \leftarrow \boldsymbol{NeighborFind}(\zeta_s, \zeta_g, \mathcal{X}_{rand})$
6:      $\mathcal{X}_s \leftarrow \boldsymbol{TreeGrow}(\mathcal{X}_{back}, \mathcal{X}_{rand})$
7:      if $\mathcal{X}_s$ not empty then
8:          $\zeta_s \leftarrow \zeta_s \cup \mathcal{X}_{rand}, \mathcal{X}_s$
9:      $\mathcal{X}_g \leftarrow \boldsymbol{TreeGrow}(\mathcal{X}_{front}, \mathcal{X}_{rand})$
10:     if $\mathcal{X}_g$ not empty then
11:         $\zeta_g \leftarrow \zeta_g \cup \mathcal{X}_{rand}, \mathcal{X}_g$
12:     $\boldsymbol{Rewire}(\zeta_s, \zeta_g, \mathcal{X}_{rand})$
13:     if $\mathcal{X}_{rand} \in \zeta_s \wedge \mathcal{X}_{rand} \in \zeta_g$ then
14:         get one solution
15: $\varGamma \leftarrow \boldsymbol{Recall}(\zeta_s, \zeta_g)$
16: $\boldsymbol{TrajectoryOptimize}(\varGamma)$
17: return $\varGamma$
```

IV. Trajectory generation

In a complex water environment, USV trajectory generation faces a real-time problem and the feasibility of trajectory execution. Based on the optimized dynamic RRT* framework, we can generate a trajectory that meets the execution of USV. Then the sampling state optimization method is added to the sampling tree expansion process to improve its speed.

A. Path searching

The main process of the dynamic RRT* framework is shown in Algorithm 1. By expanding two sampling trees ζ_s and ζ_g, feasible trajectory from $\mathcal{X}_{init}$ to $\mathcal{X}_{goal}$ is searched rapidly. In order to contain the USV nonholonomic constraints in the trajectory and improve the efficiency and stability of sampling, we combine the Dubins and A* algorithms to generate the critical topological path points $\mathcal{P}_t$, illustrated in Fig. 3. It is suitable for an under-actuated system to use the Dubins curve here due to its smoothness. Based on this, the sampling interval is constructed to reduce the sampling of outside areas effectively. The $\boldsymbol{Steer}()$ function in the $\boldsymbol{TreeGrow}()$ and $\boldsymbol{Rewire}()$ steps constructs the transfer trajectory from $\mathcal{X}_{rand}$ to sampling tree ζ_s and ζ_g that satisfies the USV dynamics. In the next part, we will introduce to use of closed form minimal value cost function to ensure stability and efficiency. If the sampling status $\mathcal{X}_{rand}$ exists in the sampling tree ζ_s and ζ_g simultaneously, then a feasible trajectory is searched.

In the traditional RRT* framework, the discarding method is adopted when the sampling state cannot be connected to the sampling tree due to obstacles or dynamic constraints. It increases the time-consuming of the algorithm. To reduce the occurrence of this situation, we use optimization method to adjust the inappropriate transfer trajectory locally so that the trajectory meets the feasibility requirements.

B. Trajectory planning with flatness

In section III-B, the differential flatness of USV has been described. Based on this system's characteristics, we only need to pay attention to the flat output $\boldsymbol{z}(t)$ and its higher derivatives when considering the planning state space of trajectory. Here, we use a linear time invariant (LTI) system to express the motion characteristics of the system, and the trajectory can be denoted as a parametric polynomial about time. The state equation of the USV system is expressed as

$$\begin{gathered}\dot{\mathcal{X}}(t) = \boldsymbol{A}\mathcal{X}(t) + \boldsymbol{B}\mathcal{U}(t), \\ \boldsymbol{A} = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{I} \\ \boldsymbol{0} & \boldsymbol{0} \end{bmatrix}, \quad \boldsymbol{B} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{I} \end{bmatrix}, \\ \mathcal{X}(t) = [\boldsymbol{p}(t), \boldsymbol{v}(t), \boldsymbol{a}(t)]^\top, \quad \mathcal{U}(t) = \boldsymbol{j}(t),\end{gathered} \tag{10}$$

where $\mathcal{X}(t)$ represents USV planning state in global-fixed coordinate frame, including position $\boldsymbol{p}(t)$, velocity $\boldsymbol{v}(t)$ and acceleration $\boldsymbol{a}(t)$ and $\mathcal{U}(t)$ is denoted as the input of LTI system. Therefore, the goal of trajectory generation is to calculate a cubic differentiable trajectory polynomial. The aim is to guide the USV from initial state $\mathcal{X}_{init}$ to final state $\mathcal{X}_{end}$ along the time, *i.e.*, the optimal boundary value problem (OBVP). Similar to [21], the following cost function needs to be optimized.

$$\mathcal{J}_{seg} = \int_0^T \left(\sigma + \frac{1}{2}\|\mathcal{U}(t)\|^2\right) dt, \tag{11}$$

where σ denotes the weight of time. Based on the cost Eq. 11, we can obtain the expression of the optimal trajectory from $\mathcal{X}_{init}$ to $\mathcal{X}_{end}$ with a closed form solution by using Pontryagin minimization method.

$$\begin{gathered}\mathcal{X}(T) = \begin{bmatrix} \frac{\alpha}{120}T^5 + \frac{\beta}{24}T^4 + \frac{\gamma}{6}T^3 + \frac{\boldsymbol{a}_0}{2}T^2 + \boldsymbol{v}_0 T + \boldsymbol{p}_0 \\ \frac{\alpha}{24}T^4 + \frac{\beta}{6}T^3 + \frac{\gamma}{2}T^2 + \boldsymbol{a}_0 T + \boldsymbol{v}_0 \\ \frac{\alpha}{6}T^3 + \frac{\beta}{2}T^2 + \gamma T + \boldsymbol{a}_0 \end{bmatrix} \\ \mathcal{U}(T) = \frac{\alpha}{2}T^2 + \beta T + \gamma,\end{gathered} \tag{12}$$

where $\mathcal{X}_{init} = [\boldsymbol{p}_0, \boldsymbol{v}_0, \boldsymbol{a}_0]^\top$ and the parameters α, β and γ can be expressed by $\mathcal{X}_{init}$ and $\mathcal{X}_{end}$, referring to [21].

Algorithm 2 *TreeGrow*

```
Input: Environment $env$, SamplingTree $\zeta$, NeighborState $\mathcal{X}_n$ SamplingState $\mathcal{X}_{rand}$
Output: ParentState $\mathcal{X}_{parent}$
 1: Initialize: $\mathcal{X}_{parent} \leftarrow null, cost_{min} \leftarrow \infty$
 2: for $\mathcal{X}_{cur}$ in $\mathcal{X}_n$ do
 3:     $\Gamma_{seg} \leftarrow \boldsymbol{Steer}(\mathcal{X}_{cur}, \mathcal{X}_{rand})$
 4:     if $\boldsymbol{FeasibilityCheck}(\Gamma_{seg}, env)$ then
 5:         if $\boldsymbol{TrajCost}(\Gamma_{seg}) < cost_{min}$ then
 6:             $\mathcal{X}_{parent} \leftarrow \mathcal{X}_{cur}$,
 7:             $cost_{min} \leftarrow \boldsymbol{TrajCost}(\Gamma_{seg})$
 8:     else
 9:         $\Gamma_{seg} \leftarrow \boldsymbol{TrajectoryOptimize}(\Gamma_{seg}, env)$
10:         if $\boldsymbol{FeasibilityCheck}(\Gamma_{seg}, env)$ then
11:             $\mathcal{X}_{parent} \leftarrow \mathcal{X}_{cur}$,
12:             $cost_{min} \leftarrow \boldsymbol{TrajCost}(\Gamma_{seg})$
13: return $\mathcal{X}_{parent}$
```

According to the form of $\mathcal{U}(T)$, Eq. 11 can be rewritten as

$$\begin{aligned}\mathcal{J}_{seg} = \sigma T + \sum_{i \in x,y} (&\frac{\alpha_i^2}{20}T^5 + \frac{\alpha_i\beta_i}{4}T^4 + \frac{\alpha_i\gamma_i + \beta_i^2}{3}T^3 \\ &+ \beta_i\gamma_i T^2 + \gamma_i^2 T).\end{aligned} \tag{13}$$

where $i \in x, y$ denotes space dimensions considered in USV trajectory generation. It can be found that $\mathcal{J}_{seg}$ is a higher-order polynomial about time T. Then the optimal trajectory can be constructed by calculating the optimal time T^* of minimum cost via solving $\mathrm{d}\mathcal{J}_{seg}/\mathrm{d}T = 0$.

The $\boldsymbol{TreeGrow}()$ process is described in Algorithm 2, where $\boldsymbol{Steer}()$ quickly obtains the optimal trajectory between two states by solving Eq. 11 in a closed form so that the algorithm can obtain the feasible trajectory. The obtained trajectory can be added into the sampling tree only when it passes the feasibility test of $\boldsymbol{FeasibilityCheck}()$.

V. Trajectory optimization and tracking

In the trajectory generation framework, trajectory feasibility can not be guaranteed. To reduce the number of samples and improve efficiency, we adjust the trajectory to meet the relevant constraints through local optimization according to the infeasible trajectory. Referring to [22], the closed form of solution can be calculated by setting the objective function to a quadratic form.

A. Optimization modelling

In order to optimize the trajectory, the constraints include the four following items, *i.e.*, trajectory smoothness, collision-free, dynamic, and original trajectory constraints. We divide the original trajectory into multiple segments based on time so that multiple free states are in the middle to prevent poor performance of whole trajectory optimization. Then, the objective function is defined as

$$\mathcal{F} = \lambda_s f_s + \lambda_c f_c + \lambda_d f_d + \lambda_o f_o, \tag{14}$$

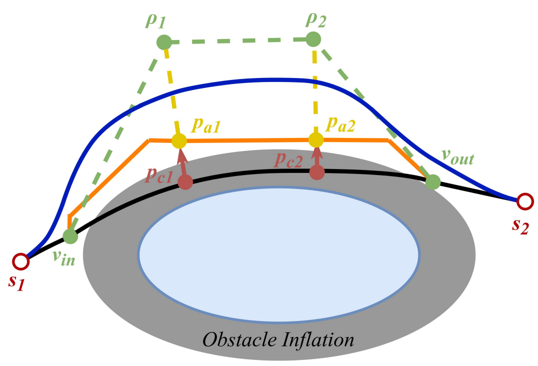

Fig. 4. An illustration of trajectory optimization. The orange path is sampled by A* algorithm from original trajectory point v_{in} to v_{out}, and then the anchor points p_a are sampled. Finally, the blue path is generated by moving p_a to the safety guidance points $\rho \in \mathcal{P}_t$.

where f_s, f_c, f_d and f_o denote the above items, and λ_s, λ_c, λ_d and λ_o are the weights respectively.

The trajectory smoothness term f_s is the time integral of the trajectory jerk, *i.e.*,

$$f_s = \int_0^T \|\mathcal{U}(t)\|^2 dt = \boldsymbol{c}^\top \boldsymbol{Q}_s \boldsymbol{c}, \tag{15}$$

where $\boldsymbol{c}^\top = [\boldsymbol{c}_1^\top, \boldsymbol{c}_2^\top, \ldots, \boldsymbol{c}_n^\top]$ is composed of coefficient vector of n trajectory segments, and $\boldsymbol{Q}_s$ is a diagonal matrix consisting of time T_i of different orders in trajectory polynomial. $T = T_1 + T_2 + \ldots + T_n$ is the total duration of trajectory and T_n is the duration of one divided piece.

Referring to [8], the collision-free term f_c is described as the integral of the distance between the safety guidance point ρ and the collision trajectory segment. The process is illustrated in Fig. 4. This makes the trajectory of the collision part tend to the safety guidance points ρ to keep away from obstacles.

$$\begin{aligned} f_c &= \sum_{\rho \in \mathcal{P}_t} \int_{T_s^\rho}^{T_e^\rho} \|\boldsymbol{p}(t) - \boldsymbol{p}_\rho(t)\|^2 dt \\ &= \sum_{\rho \in \mathcal{P}_t} (\boldsymbol{c} - \boldsymbol{c}^\rho)^\top \boldsymbol{Q}_{c,\rho} (\boldsymbol{c} - \boldsymbol{c}^\rho). \end{aligned} \tag{16}$$

The collision part is obtained by $\boldsymbol{FeasibilityCheck}()$, and the topological path is updated using A* algorithm. $\boldsymbol{p}_\rho(t)$ denotes the position of the safety guidance points ρ, and $(T_e^\rho - T_s^\rho) \subseteq [0, T]$ is the corresponding time period of collision part influenced by ρ.

The dynamic constraint term f_d limits the range of higher-order derivatives of the trajectory and punishes the trajectory part that exceeds the physical dynamics of USV. That is

$$\begin{aligned} f_d &= \sum_{\iota \in \mathcal{S}} \int_{T_s^\iota}^{T_e^\iota} \|\boldsymbol{v}(t) - \boldsymbol{v}_{max}\|^2 + \|\boldsymbol{a}(t) - \boldsymbol{a}_{max}\|^2 dt \\ &= \sum_{\iota \in \mathcal{S}} (\boldsymbol{c} - \boldsymbol{c}^\iota)^\top \boldsymbol{Q}_{d,\iota} (\boldsymbol{c} - \boldsymbol{c}^\iota), \end{aligned} \tag{17}$$

where $\boldsymbol{v}_{max}$ and $\boldsymbol{a}_{max}$ denote the maximal speed and acceleration of USV respectively. $\mathcal{S}$ is the trajectory part that exceeds the system physical limit. T_s^ι and T_e^ι indicate the start and end time of the trajectory part $\mathcal{S}$ respectively.

The original trajectory constraint term f_o is similar to the collision-free term, except that the original trajectory position information $\boldsymbol{p}^*(t)$ is used as the limit.

$$f_o = \int_0^T \|\boldsymbol{p}(t) - \boldsymbol{p}^*(t)\|^2 dt = (\boldsymbol{c} - \boldsymbol{c}^*)^\top \boldsymbol{Q}_o (\boldsymbol{c} - \boldsymbol{c}^*). \tag{18}$$

The function of term is to constrain the safe part of original trajectory to maintain its original feasibility.

Referring to [22], trajectory optimization function 14 has positive definite property, and the optimal solution can be obtained through a closed form. When the trajectory still does not satisfy the requirements, the unsatisfactory segmentation will be extracted, and the weight of the corresponding optimization term is modified with a more significant impact so that the trajectory meets all constraints after multiple iterations.

In our framework, the trajectory generated by the front end based on dynamics is composed of several segments, so its overall smoothness can not be guaranteed, which is not optimal for tracking. Therefore, these trajectory segments need to be optimized for trajectory smoothness. We use the same optimization function as above, except that the smoothness weight λ_s is enlarged and dynamic constraint weight λ_d is reduced. Meanwhile, because there is no collision in the optimized trajectory, we directly set the original trajectory as the safety route $\mathcal{P}_t$ to extract guidance point ρ when constructing the collision-free term f_c. After smooth optimization, the trajectory is more feasible and stable for USV while satisfying other constraints.

B. Motion tracking with NMPC

The safe and feasible trajectory generated by the planning module needs effective trajectory tracking control to realize the autonomous navigation of USV. To make USV have stable tracking ability in complex water environments, we set a high control frequency to 100Hz, and contain all the states $\boldsymbol{x}$ into the control law.

The nonlinear model predictive control (NMPC) method is selected as the controller. According to the general form of MPC, a quadratic optimization problem is constructed about the error between the given trajectory state and the forward prediction state. The following nonlinear discrete optimization problem is iteratively solved.

$$\begin{aligned} \min_{\substack{\boldsymbol{x}_0, \ldots, \boldsymbol{x}_N, \\ \boldsymbol{u}_0, \ldots, \boldsymbol{u}_{N-1}}} \; & \Delta\boldsymbol{x}_N^\top \boldsymbol{Q}_N \Delta\boldsymbol{x}_N + \sum_{i=0}^{N-1} \Delta\boldsymbol{x}_i^\top \boldsymbol{Q} \Delta\boldsymbol{x}_i + \boldsymbol{u}_i^\top \boldsymbol{R} \boldsymbol{u}_i \\ s.t. \quad & \boldsymbol{x}_{i+1} = \boldsymbol{f}_{RK4}(\boldsymbol{x}_i, \boldsymbol{u}_i, \Delta t) \\ & \boldsymbol{x}_0 = \boldsymbol{x}_{init}, \boldsymbol{u}_{min} \le \boldsymbol{u}_i \le \boldsymbol{u}_{max} \\ & \Delta\boldsymbol{x}_i = \boldsymbol{x}_i - \boldsymbol{x}_{i,ref}, \end{aligned} \tag{19}$$

where $\boldsymbol{x}_0$ is the actual state of USV at each solution cycle time, $\boldsymbol{x}_{i,ref}$ represents a reference obtaining from trajectory generation, the forward prediction time T_f is discretized into

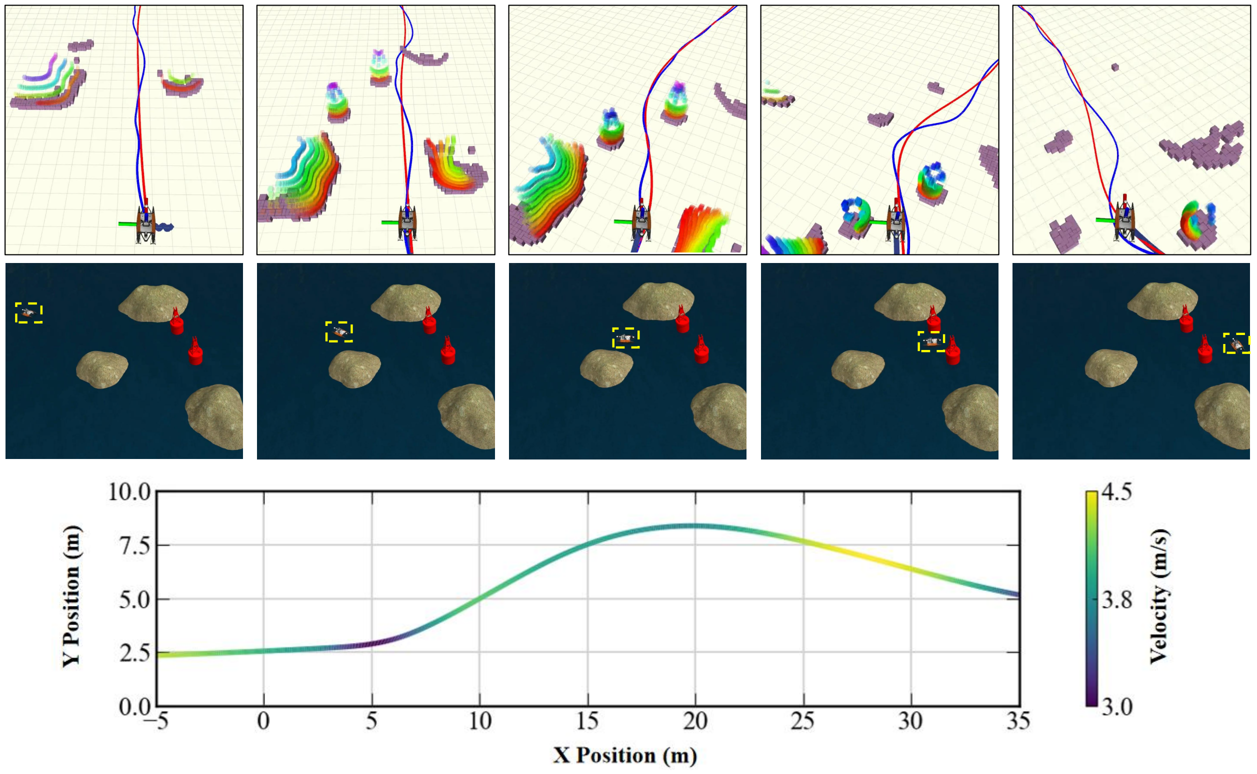

Fig. 5. Visualization of trajectory planning over a short period of time with velocity profile.

N time steps, and $\boldsymbol{Q}_N$, $\boldsymbol{Q}$, $\boldsymbol{R}$ are weighting matrices.The weights were selected experimentally as

$$\begin{aligned} \boldsymbol{R} &= diag([5,5]), \\ \boldsymbol{Q} &= diag([15,15,7,5,1,1]), \\ \boldsymbol{Q}_N &= diag([30,30,15,10,2,2]). \end{aligned} \tag{20}$$

This optimization problem can be transformed into a sequential quadratic programming (SQP) problem, and the control process is implemented through Acados library [23].

VI. Simulation Results

In this section, we perform simulation experiments using the open source USV simulator Otter [24] to verify the proposed method within the ROS environment.

A. Simulation environment

The Otter USV is a catamaran cloned from the real world, and we have slightly adjusted some physical parameters of the USV according to the scenario in the simulation experiments. It has a size of 1.53m long, 1.08m wide, and 0.82m high, with a total mass of about 29kg, and each propeller can provide a bidirectional thrust of the maximum 100N. Sensing equipment, including LIDAR, camera, and GPS, have been equipped in the simulator. The Otter USV sails in a rugged island environment with various floating obstacles.

The experimental scenario is constructed within Gazebo, and the scene size is about 300×300 square meters. The environment includes random known and unknown obstacles. Moreover, the scene contains a constant disturbance wind field and wave that causes the USV to swing and drift. All experiments are conducted on a laptop with an Intel i7-1165G7 CPU and a Linux platform running ROS.

In order to identify the unknown parameters $\boldsymbol{h}$ in the Otter USV, we use random input and sinusoidal input respectively to generate two different kind of USV system motion data for solving Eq. 8. Eventually, the unknown parameters for the Otter USV are as $\boldsymbol{h} = [m_1, m_2, d_1, d_2, d_3] = [38.5, 14.5, 19.4, 20.5, 18.6]$.

B. Path planning analysis

As shown in Fig. 5, the Otter USV travels across a complex island with various unknown obstacles. The path planning module can generate an obstacle-avoidance trajectory in real-time. The blue line shows the planning results by dynamic RRT*, which is random and not optimal. After the optimization process proposed by our method, the USV can obtain a smooth and kinetic affine trajectory to cross all kinds of terrain in the best posture.

The ablation study for the proposed method is shown in Table I, where the baseline denotes the dynamic RRT* algorithm, and local optimization (+Local Opt.) means trajectory local adjustment with differential flatness, and global optimization (+Global Opt.) means the trajectory optimization with dynamic constraints. Trajectory length, time, and cost are associated with generated trajectory. Node utilization is the ratio of planning algorithm utilization to planned nodes and total sampling nodes, which is used to evaluate the

TABLE I

ABLATION STUDY FOR THE PROPOSED METHOD.

Method	Traj. Len. (m)	Traj. Time (s)	Traj. Cost -	Node Util. (%)	Algo. Time (ms)	Succ. Rate (%)
Baseline	71.4	**20.5**	24.3	29.7	58.1	86
+Local Opt.	71.8	20.8	24.2	40.6	47.8	96
+Global Opt.	69.6	21.6	22.1	30.8	60.2	84
Proposed	**69.3**	21.3	**21.9**	**41.8**	**45.6**	**98**

algorithm's efficiency. Algorithm time and success rate are used to measure the performance of the algorithm. The algorithm is tested repeatedly 50 times, and one successful planning should be completed within $100ms$.

In the baseline model test results, the node sampling rate is obviously low (29.7%), and the problem of calculation timeout frequently occurs, resulting in a low success rate (86%). At the same time, the algorithm consumes more time ($58.1ms$). On the other hand, the length of trajectory generated by the baseline model is long, and the trajectory is not smooth. This also can be viewed from Fig. 5.

After adding the local optimization method, the node utilization, planning time, and success rate have been significantly improved (40.6%, $47.8ms$, 96%), which indicates that it has a significant effect on the efficiency and stability of the planning algorithm. After the global optimization method is added, the path length and cost are reduced because it makes the path smoother, and the algorithm time consumption does not increase too much.

As for the proposed method, we combine the baseline model with local and global optimization, resulting in a smoother trajectory and higher planning efficiency. The trajectory length and cost are the lowest ($69.3m$, 21.9), while the node utilization, algorithm time, and successful rate achieve the best (41.8%, $45.6ms$, 98%).

Taking a closer look at the state variables of the planned trajectory as shown in Fig. 6, there are few shocks in the surge and sway velocity curves, as well as the yaw and its rate. As for the control inputs, the desired thrusts are generally stable, which shows that the control performance requirements for the USV are relatively low in this case.

C. Motion control analysis

As for the motion control performance, the results are illustrated in Fig. 7. The proposed control method will track two trajectory types, including the spiral type and splayed shape trajectory. Meanwhile, we test the control performance with and without disturbance. From the results, we can see that the tracking control performance remains at a high level. Overall, the accurate trajectory of USV is consistent with the planned trajectory, and the tracking error is controlled within 0.4m in a 300×300 square meters simulation environment. The tracking velocity is quite large, ranging up to about 4m/s, which is quite a large velocity for a small USV.

The quantitative comparisons between the baseline model (NMPC without DF) and the proposed method are summarized in Table II. Although the USV trajectory tracking

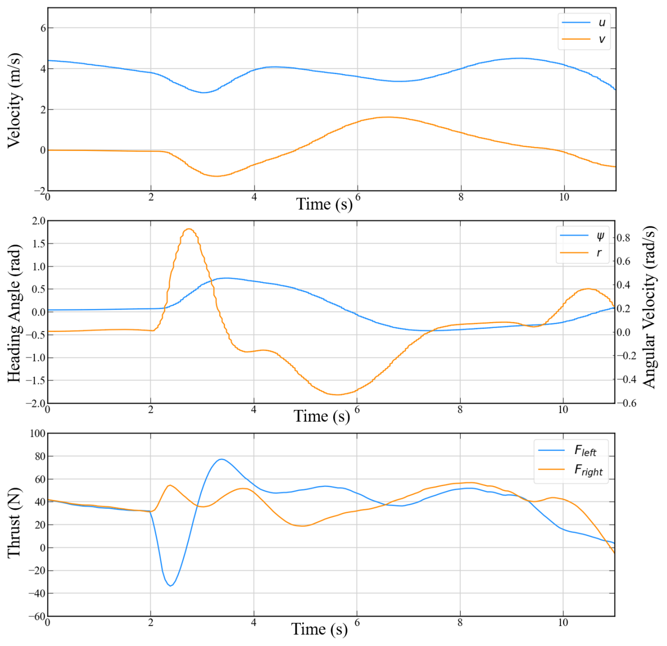

Fig. 6. State variables with time for the planned trajectory.

TABLE II

ABLATION STUDY FOR DIFFERENTIAL FLATNESS (DF) IN MOTION CONTROL.

Method	Mean Error (m)	Max Error (m)	Mean Vel. (m/s)	Max Vel. (m/s)	Ang. Vel. Integral (rad^2/s)
w.o DF	0.257	0.768	2.93	4.31	2.01
Ours	**0.121**	**0.483**	**3.32**	**4.37**	**0.95**

algorithm can still track the trajectory in real-time without differential flatness, the control effect is poor, the navigation is not smooth, and the mean tracking error achieves $0.257m$. After adding differential flatness, the angular velocity integral has significantly decreased (from 1.97 to 0.82), which indicates that differential flatness makes a better connection between path planning and motion control components. It has a better improvement in the robustness of motion control.

VII. CONCLUSION

In this paper, we propose a novel yet efficient trajectory planning and motion control algorithm for USV. Traditional methods use trajectory designed for UAV or AGV as the guidance to generate an obstacle-avoidance path, which ignores the inherent under-actuated characteristics of USV. Based on this, we propose a trajectory planning algorithm that concerns the hull dynamics during the process and uses differential flatness to improve algorithm efficiency. The generated trajectory is in good agreement with the characteristics of the hull, so the control effect remains at a high level. We perform extensive simulation experiments to verify the effectiveness of the proposed method in planning and controlling performance for USV. In the future, we will

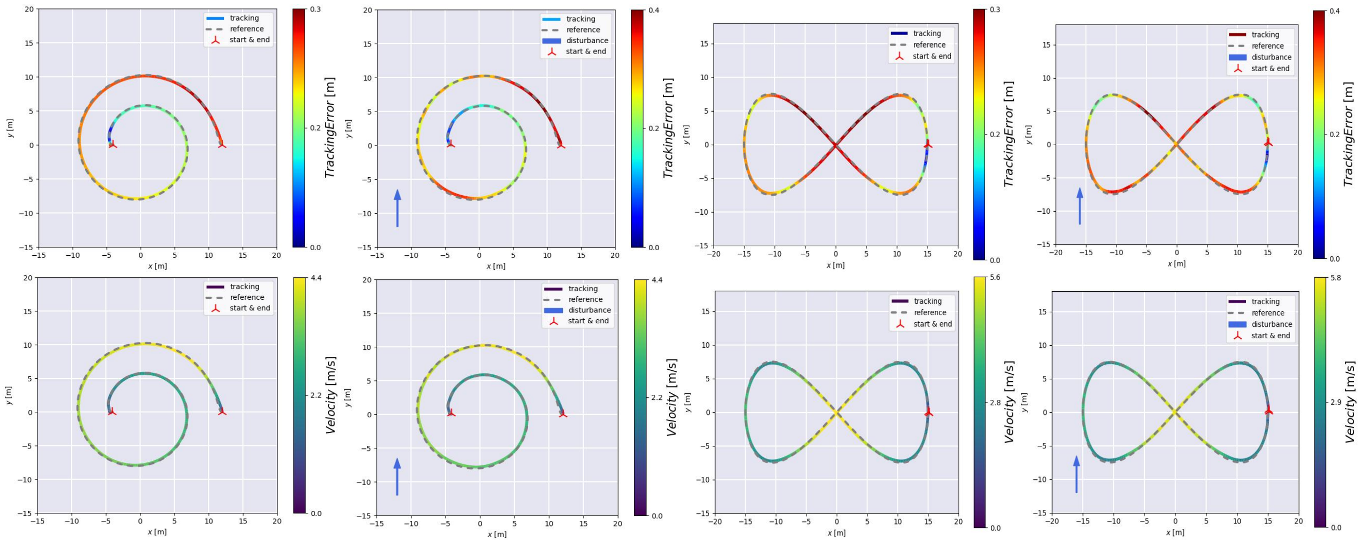

Fig. 7. Visualization of motion tracking results of a spiral type and splayed shape trajectory with and without disturbances.

carry out high-precision system identification on a real ship and realize it in real objects.

References

[1] Y. Qiao, J. Yin, W. Wang, F. Duarte, J. Yang, and C. Ratti, "Survey of deep learning for autonomous surface vehicles in marine environments," *IEEE Transactions on Intelligent Transportation Systems*, 2023.

[2] Z. Chen, T. Huang, Z. Xue, Z. Zhu, J. Xu, and Y. Liu, "A novel unmanned surface vehicle with 2d-3d fused perception and obstacle avoidance module," in *2021 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2021, pp. 1804–1809.

[3] W. Wang, T. Shan, P. Leoni, D. Fernández-Gutiérrez, D. Meyers, C. Ratti, and D. Rus, "Roboat ii: A novel autonomous surface vessel for urban environments," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 1740–1747.

[4] L. Wen, J. Yan, X. Yang, Y. Liu, and Y. Gu, "Collision-free trajectory planning for autonomous surface vehicle," in *2020 IEEE/ASME international conference on advanced intelligent mechatronics (AIM)*. IEEE, 2020, pp. 1098–1105.

[5] T. I. Fossen, *Handbook of marine craft hydrodynamics and motion control*. John Wiley & Sons, 2011.

[6] L. E. Dubins, "On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents," *American Journal of mathematics*, vol. 79, no. 3, pp. 497–516, 1957.

[7] Z. Du, Y. Wen, C. Xiao, L. Huang, C. Zhou, and F. Zhang, "Trajectory-cell based method for the unmanned surface vehicle motion planning," *Applied Ocean Research*, vol. 86, pp. 207–221, 2019.

[8] Z. Wang, X. Zhou, C. Xu, and F. Gao, "Geometrically constrained trajectory optimization for multicopters," *IEEE Transactions on Robotics*, vol. 38, no. 5, pp. 3259–3278, 2022.

[9] B. Mu and P. Chirarattananon, "Trajectory generation for underactuated multirotor vehicles with tilted propellers via a flatness-based method," in *2019 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*. IEEE, 2019, pp. 1365–1370.

[10] S. Han, L. Wang, Y. Wang, and H. He, "A dynamically hybrid path planning for unmanned surface vehicles based on non-uniform theta* and improved dynamic windows approach," *Ocean Engineering*, vol. 257, p. 111655, 2022.

[11] K. Yu, X.-f. Liang, M.-z. Li, Z. Chen, Y.-l. Yao, X. Li, Z.-x. Zhao, and Y. Teng, "Usv path planning method with velocity variation and global optimisation based on ais service platform," *Ocean Engineering*, vol. 236, p. 109560, 2021.

[12] X. Xu, P. Cai, Z. Ahmed, V. S. Yellapu, and W. Zhang, "Path planning and dynamic collision avoidance algorithm under colregs via deep reinforcement learning," *Neurocomputing*, vol. 468, pp. 181–197, 2022.

[13] G. Zhang, J. Han, J. Li, and X. Zhang, "Apf-based intelligent navigation approach for usv in presence of mixed potential directions: Guidance and control design," *Ocean Engineering*, vol. 260, p. 111972, 2022.

[14] G. X. Wu, Y. Ding, T. Tahsin, and I. Atilla, "Adaptive neural network and extended state observer-based non-singular terminal sliding mode-tracking control for an underactuated usv with unknown uncertainties," *Applied Ocean Research*, vol. 135, p. 103560, 2023.

[15] B. Du, B. Lin, C. Zhang, B. Dong, and W. Zhang, "Safe deep reinforcement learning-based adaptive control for usv interception mission," *Ocean Engineering*, vol. 246, p. 110477, 2022.

[16] E. I. Sarda, I. R. Bertaska, A. Qu, and K. D. von Ellenrieder, "Development of a usv station-keeping controller," in *OCEANS 2015-Genova*. IEEE, 2015, pp. 1–10.

[17] S. Han, L. Wang, Y. Wang, and H. He, "An efficient motion planning based on grid map: Predicted trajectory approach with global path guiding," *Ocean Engineering*, vol. 238, p. 109696, 2021.

[18] R. Morales, H. Sira-Ramírez, and J. Somolinos, "Linear active disturbance rejection control of the hovercraft vessel model," *Ocean Engineering*, vol. 96, pp. 100–108, 2015.

[19] J. A. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "Casadi: a software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, 2019.

[20] M. Fliess, J. Lévine, P. Martin, and P. Rouchon, "Flatness and defect of non-linear systems: introductory theory and examples," *International journal of control*, vol. 61, no. 6, pp. 1327–1361, 1995.

[21] B. Zhou, F. Gao, L. Wang, C. Liu, and S. Shen, "Robust and efficient quadrotor trajectory generation for fast autonomous flight," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3529–3536, 2019.

[22] C. Richter, A. Bry, and N. Roy, "Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments," in *Robotics research*. Springer, 2016, pp. 649–666.

[23] R. Verschueren, G. Frison, D. Kouzoupis, J. Frey, N. v. Duijkeren, A. Zanelli, B. Novoselnik, T. Albin, R. Quirynen, and M. Diehl, "acados—a modular open-source framework for fast embedded optimal control," *Mathematical Programming Computation*, vol. 14, no. 1, pp. 147–183, 2022.

[24] J. H. Lenes, "Autonomous online path planning and path-following control for complete coverage maneuvering of a usv," Master's thesis, NTNU, 2019.