Global path planning and waypoint following for heterogeneous unmanned surface vehicles assisting inland water monitoring

Liang Zhao [a], Yong Bai [a, *], Jeom Kee Paik [b, c]
[a] College of Civil Engineering and Architecture, Zhejiang University, Hangzhou 310058, People's Republic of China
[b] School of Maritime and Transportation, Ningbo University, Ningbo, People's Republic of China
[c] Department of Mechanical Engineering, University College London, London, the United Kingdom
Correspondence: Yong Bai, baiyong@zju.edu.cn

Abstract

The idea of dispatching multiple unmanned surface vehicles (USVs) to undertake marine missions has ignited a burgeoning enthusiasm on a global scale. Embarking on a quest to facilitate inland water monitoring, this paper presents a systematical approach concerning global path planning and path following for heterogeneous USVs. Specifically, by capturing the heterogeneous nature, an extended multiple travelling salesman problem (EMTSP) model, which seamlessly bridges the gap between various disparate constraints and optimization objectives, is formulated for the first time. Then, a novel Greedy Partheno Genetic Algorithm (GPGA) is devised to consistently address the problem from two aspects: (1) Incorporating the greedy randomized initialization and local exploration strategy, GPGA merits strong global and local searching ability, providing high-quality solutions for EMTSP. (2) A novel mutation strategy which not only inherits all advantages of PGA but also maintains the best individual in the offspring is devised, contributing to the local escaping efficiently. Finally, to track the waypoint permutations generated by GPGA, control input is generated by the nonlinear model predictive controller (NMPC), ensuring the USV corresponds with the reference path and smoothen the motion under constrained dynamics. Simulations and comparisons in various scenarios demonstrated the effectiveness and superiority of the proposed scheme.

Keywords: Path planning; Unmanned surface vehicles; Water monitoring; Genetic algorithm

Note to practitioners: This paper is motivated by our ongoing experiments on the USV-assisted inland water monitoring missions, which collect monitoring data for a wide range of marine elements including water temperature, depth, salinity, biological indexes, and bathymetry. With the aid of the USVs, the collection of monitoring data benefits from a great loss of manpower and resources. However, the implementation of our approach has encountered a practical challenge due to the diverse sensors equipped on the USVs. This has resulted in certain areas being accessible only to specific USVs since they are equipped with the required sensors. For instance, the targets that required temperature or salinity data can only be visited by the USVs equipped with conductivity-temperature-depth profile collector. Unfortunately, existing literature on global path planning mainly focuses on homogeneous USVs, whereby the heterogeneous capabilities are omitted. More to the problem, after planning the target sequence, the USVs' practical waypoint following is challenging. Traditional waypoint following methods such as LOS-based is impossible to theoretically impose any constraint on the existing LOS laws, neither on the control actions nor their increments. What's more, they have also shown weak ability for disturbance rejection, which may hinder their practical applications. To address the abovementioned problems, we propose a novel systematic approach that combines global path planning and path following. Using the proposed scheme, global planning optimality under heterogeneous constraints and appropriately maneuvering the USV with a desired response within various physical constraints can be achieved simultaneously. We believe our work could benefit the readers who are currently conducting research in deploying multi-agent systems for real-world engineering problems.

1. Introduction

With artificial intelligence at the helm, the advancements of Unmanned Surface Vehicles (USVs) have been propelled to new heights, charting a course towards a brighter future of autonomous exploration and unlocking the secrets of aquatic world [1–5]. Specifically, much attention has been given to exploit USVs to perform ocean and marine tasks in hostile or human-inaccessible areas, e.g., maritime patrolling, coastal guarding, and maritime search and rescue [6–8]. Moreover, USVs can be applied to measure the environment data of inland water, which is crucial for achieving environmental sustainability and securing water resources [9–16]. The water monitoring mission follows different purposes, such as radioactive material detection [17], measuring basic marine elements (currents, temperature, and salinity) [18,19], biological investigations [20], bathymetry surveying [21], as well as observing water columns or warming trend to reveal ocean carbon cycle [20,22,23]. In order to successfully complete missions for water monitoring, the planning and control algorithms of USVs have always been the keys to such problems [24]. On the one hand, the global planning algorithm aims to optimize the path sequence by assigning task points to multiple USVs in a manner that minimizes energy consumption and equalizes the workload distribution among USVs [25,26]. On the other hand, the waypoint following control method guides the USVs to successfully visit the task points [27,28]. Therefore, to ensure efficient and reliable operation for water monitoring, the global path planning and path following problem should be addressed properly.

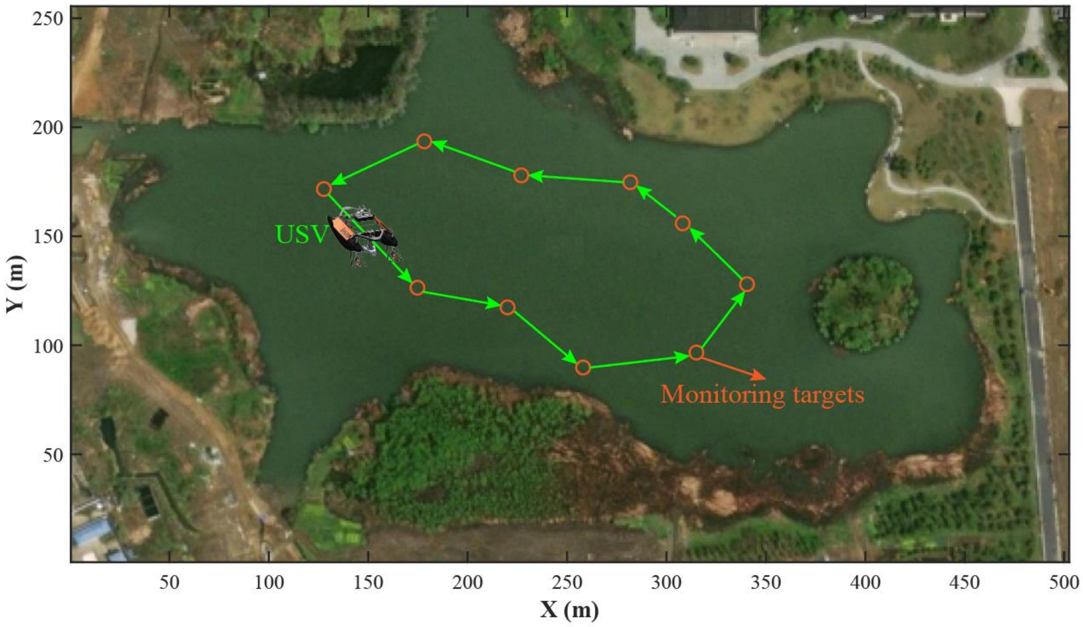

Fig. 1. Illustration of a typical monitoring mission

Global path planning for USVs aims to compute the optimal routes based on the distribution of monitoring targets, requirements of the mission, and settings of environment surroundings [6], see Fig. 1. Prior research has traditionally equated global path planning problem with the classical traveling salesman problem (TSP), wherein a set of mission targets are equally prioritized for visitation and the objective is to determine the shortest possible sequence of waypoints [25,29]. The Multiple Traveling Salesmen Problem (MTSP), which can be defined as finding the shortest route for multiple USVs, is introduced when there are various USVs involved. TSP-variants are usually non-deterministic polynomial (NP-hard) problems [26,30]. These problems are for which, even in theory, no shortcut or algorithm is possible to lead to a fast and optimal solution. To obtain an optimal solution, an exhaustive analysis of all possible outcomes is required, which is computationally intensive. Consequently, heuristic approaches such as evolutionary algorithms (EA), ant colony optimization (ACO), and particle swarm optimization (PSO) are ideal for addressing these problems since they can provide satisfactory sub-optimal solutions with comparatively low computational burden [25,31].

Presently, booming academic and technological advancements pertaining to the global path planning of USVs have emerged in the latest research works. Considering the distribution of the targets, [25] used an orientation angle-based grouping strategy to enhance PSO for water quality detection and sampling. Compensating for the inherent shortcomings of conventional GA including slow convergence and premature, [31] proposed the multiple offspring GA for the global path planning of unmanned surface vehicles. To navigate a USV in a real maritime environment, a series of studies on the implementation of improved particle swarm optimization have been carried out by [30,32]. By minimizing the energy consumption per unit time in multiple task locations, a chaotic and sharing-learning particle swarm optimization (CSPSO) algorithm is proposed [33]. To solve the multiple-waypoint path planning for survey USVs, a discrete group teaching optimization algorithm (DGTOA) is devised by [34]. Enhancing the global search ability for unmanned surface vessel path planning, [26] devised an improved differential evolution particle swarm optimization algorithm (DePSO). In conjunction with self-organizing map (SOM), an improved genetic algorithm is studied by [35] to address the path-planning problems for a multiple unmanned surface vehicle (USVs) system. However, MTSP-variants are difficult to solve since they are non-convex [36]. Existing meta-heuristic methods feature low convergence speed and may also fall into local optimization easily. Thus, a more effective combinatorial optimization method should be devised for the path planning problem, with an emphasis on improving the global search capability through the integration of a convenient and effective mechanism [36].

Moreover, one crucial aspect that has rarely been addressed by current studies is the heterogenous nature pertaining to the USVs' capabilities. In essence, the abovementioned MTSP-variant is an abstraction of the practical problems in which multiple executing individuals (homogeneous agents) are involved and share a common workspace (target points) [25,31,32,30,33,34]. However, individuals have the same workspace in real-world problems. In some cases, the targets of individual USVs are not the same but overlap with each other. Thus, each USV has to perform not only the common tasks that can be accessed by any of them but also complete the tasks that correspond to their exclusive capabilities. In water monitoring missions, since the USVs are equipped with different types of sensors, some areas should be only visited by one specific type of USV. For instance, the targets that require temperature or salinity data can only be visited by the USVs equipped with conductivity-temperature-depth profile collector. Such a problem is frequently encountered in real-world applications, yet there is quite limited research available on it. To the best of our knowledge, the global path planning problem of heterogeneous USVs is still an open and vital topic at the current stage.

In addition to global path planning, a feasible waypoint tracking strategy ensures the USV to access the reference target sequence as precisely as possible, thereby contributing to successfully completing the missions [28,37]. Waypoint tracking is similar to the straight-line path following problem, there are three objectives: 1) minimizing the cross-tracking error along the reference path and real trajectory; 2) achieving smooth turns and avoiding drastic maneuvers; and 3) maintaining a constant surge speed [38,39]. Previous studies have divided the traditional path following methods into two separate modules in a cascade structure: the guidance module and a low-level controller [37,38]. On the one hand, the guidance module is in charge of producing the set points for the heading angle and forward speed along with their corresponding time dependencies, such that the USV should follow the desired path and adhere to the time restrictions for the desired forward speed. The low-level module, on the contrary, has a controller that works with the propellers to track the set points that the guidance layer provides. As a result, in the conventional path following problem, the low-level controller concentrates on the dynamics while the guidance module concentrates on the kinematics [27,37].

In the literature, many different strategies have been proposed for the path following of USVs. For the guidance module, a well-known method for path following of straight lines is the line-of-sight (LOS) guidance, which is based on the approach of experienced helmsmen who steer the vessel toward a point lying at a constant distance ahead of the ship along the desired path. LOS has been enhanced over the years, including application to Dubins paths [40], compensating for the drift effect [41], rejection of severe ocean disturbances [42], combination with fuzzy logic system [43], and solving the large curvature path following [44]. As for the low-level control module, extensive research has taken place in the past using ideas from almost all branches of control engineering: robust control [38], sliding mode control [45], deep reinforcement learning and neural network [46–48], and backstepping control [49]. However, traditional control strategies are usually limited by the constraints on states as well as their increments in real mechanical system, and none of the above-mentioned works has considered the dynamic bounds explicitly. Moreover, since traditional path following schemes used to maneuver the USV along the prescribed path are designed separately, it is difficult to theoretically impose such dynamic limits on the traditional methods, neither on the control actions nor on their derivatives [27,37].

As observed from the foregoing works, domestic and foreign researchers undertook a series of studies on the global path planning of USVs and path follow problems. However, it should be noted that past research has certain shortcomings: (1) Since existing literature mainly focuses on the global path planning problem of homogeneous USVs, a general problem model for the heterogeneous USVs is urgently desirable; (2) We are of the opinion that there is still room for improvements in combinatorial optimization approaches to solve the non-convex problems such as MTSP-variants. A more effective combinatorial optimization method needs to be specifically designed to facilitate the solutions. (3) Traditional guidance and control schemes are separated dynamics, which is impossible to theoretically impose any constraint, neither on the input signals nor on their control increments.

Motivated by the considerations mentioned above, this paper explores the global path planning for heterogeneous USVs, and their path follow problems in the context of the water monitoring mission. The main contributions are illustrated as follows:

- A novel global path planning and waypoint following framework is proposed to formulate path-planning and path-tracking in an organically way. Augmented practicability has been achieved by extensive simulation and experimental evaluations under complex environments.
- An extended multiple travelling salesmen problem (EMTSP) is established by bridging the heterogeneous nature and various disparate constraints jointly, providing a systematic model for the global path planning of multiple heterogeneous USVs.
- Incorporating the greedy randomized initialization and local exploration, we propose the Greedy Partheno Genetic Algorithm (GPGA) to consistently address the global path planning. GPGA merits strong global searching ability and facilitates local escaping simultaneously. In such a case, the underlying optimization problem is fully exploited, and it converges quickly to generate optimal waypoint sequence.
- With the aid of the proposed nonlinear model predictive controller, reference targets can be properly tracked by virtue of the NMPC strategy where robust maneuvering is ensured by respecting USV's physical constraints and external disturbances, thereby contributing to the successful completion of water monitoring.

The remaining sections of this article are organized as follows. The problem formulation is described in Section 2. Sections 3 presents the global path planning algorithm and NMPC design. The superiority and efficiency of the proposed framework is verified through illustrative simulations in Sections 4. Finally, the concluding remarks are given in Section 5.

2. Problem formulation

The overall framework of the problem consists of two modules, i.e., extended multiple travelling salesman problem (EMTSP) and path following problem. The first module aims to obtain a multi-target cruise permutation, which provides USVs with a sequence traversing all non-repeating targets. In this process, the heterogeneity of the targets and USVs is considered. Based on the planned target sequence, the second module guides the USVs traversing all target points through an ocean environment while keeping the tracking error as small as possible.

2.1.Heterogeneous global path planning problem

2.1.1. USV model

Suppose the set of the USVs is denoted by $U_k = \{U_1, U_2, U_3, \ldots, U_{N_U}\}$, $k = 1, 2, 3, \ldots, N_U$, and N_U is the number of the USVs. Due to the various types of the equipment onboard, the first attribute lies on the functionality of the USVs. Suppose the USV has the attribute of the exclusive functional type, which is denoted by $F^k = \{F_1, F_2, \ldots, F_{N_F}\}$, $k = 1, 2, 3, \ldots, N_U$, where N_F is number of the types. It indicates that USV U_k possesses the unique capability of executing a specific type of task, e.g., the mapping mission must be performed by the USVs with surveying devices onboard while the attacking mission must be completed by

USVs with weapons.

2.1.2. Task model

Suppose the set of the tasks is denoted by $T_i = \{T_1, T_2, T_3, \dots, T_{N_T}\}$, and N_T is the number of the targets. To be in accordance with the USVs' functions, the task set is divided into $N_F + 1$ disjoint nonempty sets, i.e., tasks with common functional type $F_C{}'$ and tasks with exclusive functional type F', $\forall F' \in \{F_1', F_2', \dots, F_{N_F}'\}$. The common tasks can be visited by any USVs while the exclusive tasks can be only accessed by specific USVs, and it is formulated as follows:

$$F^k = F^{i\,\prime} \text{ or } F^k = F_c^{i\,\prime},\ i \in 1,2,\dots,N_T,\ k = 1,2,\dots,N_U. \tag{1}$$

2.1.3. Problem statement

To determine the task sequence, an extended multiple traveling salesman problem (EMTSP) is formulated, in which the heterogeneous nature is considered. Suppose the set of the USVs is denoted by $U_k = \{U_1, U_2, U_3, \dots, U_{N_U}\}$, and the target set is denoted as $T_i = \{T_1, T_2, T_3, \dots, T_{N_T}\}$, where $N_U < N_T$. It can be formulated over a complete digraph $G(\mathfrak{H}, E)$, where vertex set $\mathfrak{H} = \{0, 1, 2, \dots, N_T - 1\}$ numbers the tasks; and each edge in $(i,j) \in E$, $i \neq j$, is associated with a weight ω_{ij} representing a visit cost between two tasks i and j. The binary variable $x_{ijk} = 1$, $i \neq j$, $i,j \in \mathfrak{H}$, $k \in Z$, if the kth USV passes through edge (i,j); and otherwise $x_{ijk} = 0$. Consequently, the tour cost ω_{ij} is obtained by calculating the distance between tasks i and j, which gives:

$$\omega_{i,j} = \|T_i - T_j\|,\ (i,j) \in E. \tag{2}$$

Then the total cost of the USV U_k is:

$$D_k = \sum_{i=0}^{N_T-1} \sum_{j=0}^{N_T-1} \omega_{i,j} x_{ijk}, \tag{3}$$

Based on the aforementioned models, the formulated multi-objective problem is stated as follows:

$$\min F = f_1 + f_2 \tag{4}$$

$$f_1 = \textstyle\sum_{k=1}^{N_U} D_k \tag{5}$$

$$f_2 = \underbrace{\mathrm{Max}\, D_k}_{k=1,2,\dots,N_U} - \underbrace{\mathrm{Min}\, D_l}_{l=1,2,\dots,N_U} \tag{6}$$

subject to the following constraints:

$$\omega_{i,j} = \|T_i - T_j\|,\ (i,j) \in E \tag{7}$$

$$D_k = \textstyle\sum_{i=0}^{N_T-1} \sum_{j=0}^{N_T-1} \omega_{i,j} x_{ijk} \tag{8}$$

$$\textstyle\sum_{i=1}^{N_T} x_{0ik} = 1,\ i \in \mathfrak{H},\ k = 1,2,\dots,N_U \tag{9}$$

$$\textstyle\sum_{i=1}^{N_T} x_{j0k} = 1,\ j \in \mathfrak{H},\ k = 1,2,\dots,N_U \tag{10}$$

$$\textstyle\sum_{k=1}^{N_U} \sum_{i=0}^{N_T-1} x_{ijk} = 1,\ i \neq j,\ j \in \mathfrak{H}\backslash\{0\},\ k = 1,2,\dots,N_U, \text{if } \alpha = 0 \tag{11}$$

$$F^k = F^{i\,\prime} \text{ or } F^k = F_c^{i\,\prime},\ i \in \mathfrak{H},\ k = 1,2,\dots,N_U. \tag{12}$$

Remark. 1. The constraints are expounded as follows. Eq. (7)-(8) denote the expressions of the visit cost $\omega_{i,j}$ and total cost of a USV D_k. Eq. (9)-(10) indicate the every USV starts from and returns to the depot after the tour. Eq. (11) denotes each task except depot must be visited by a USV exactly once. Eq. (12) indicates that the USV must visit the its own exclusive task or a common task.

2.2.Path following problem

This section briefly describes the three-DOF maneuvering model for the motion of an USV moving in the horizontal plane and a basic statement of the path following problem. For more details, the reader is referred to [50].

2.2.1. Basic assumptions

The general model of a typical USV has six degree-of-freedom (DOF): surge, sway, yaw, heave, roll, and pitch. These can be simplified into a 3-DOF model with the following assumptions:

Assumption 1: The motions that generated by wind, waves, and currents including heave, roll, and pitch are negligible.
Assumption 2: The hydrodynamic damping is linear.
Assumption 3: The control actions consist of surge force and yaw moment.
Assumption 4: The inertia-related and damping-related matrices are diagonal [51].

Remark 2. Nonlinear damping is not considered, since it would increase the complexity of the controller without contributing to improving the result.

2.2.2. Vessel model

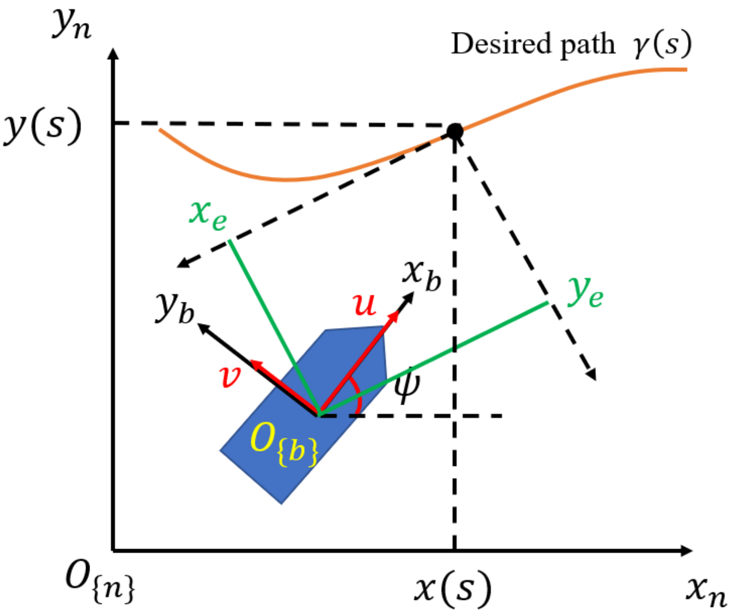

Fig. 2. Geometry of the coordinate system

Based on the assumptions, the 3 DOF kinematic and dynamic model of a surface vessel in a horizontal plane (see Fig. 2) is:

$$\begin{gathered}\dot{\boldsymbol{\eta}} = \boldsymbol{R}(\psi)\boldsymbol{v}_r \\ \boldsymbol{M}\dot{\boldsymbol{v}}_r + \boldsymbol{C}(\dot{\boldsymbol{v}}_r)\dot{\boldsymbol{v}}_r + \boldsymbol{D}\boldsymbol{v}_r = \boldsymbol{\tau},\end{gathered} \tag{13}$$

where $\boldsymbol{\eta} = [x, y, \psi]^T$ denotes the position coordinates and heading angle in the earth-fixed inertial frame {n}, $\boldsymbol{v}_r = [u_r, v_r, r]^T = \boldsymbol{v} - \boldsymbol{v}_c$ includes the relative velocities in the body-fixed frame {b}, $\boldsymbol{\tau} = [\tau_u, 0, \tau_r]^T$ gathers the vector of control signals. It is worth to mention that the underactuated configuration is considered in this paper since the surge force and yaw moment are the only control forces. The rotation matrix $\boldsymbol{R}(\psi)$ denotes the transformation between the body-fixed frame and the earth-fixed inertial frame:

$$\boldsymbol{R}(\psi) = \begin{bmatrix} cos\ (\psi) & -sin\ (\psi) & 0 \\ sin\ (\psi) & cos\ (\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{14}$$

The mass matrix $\boldsymbol{M} = \boldsymbol{M^T} > 0$ includes the inertial features of the USV and hydrodynamic added mass. The matrix $\boldsymbol{D}$ includes the damping coefficients. The Coriolis matrix $\boldsymbol{C}$, which includes the Coriolis and centripetal effects, can be derived from $\boldsymbol{M}$. According to the forementioned assumptions, the matrices $\boldsymbol{M}$, $\boldsymbol{C}$, and $\boldsymbol{D}$ can be expressed as:

$$\boldsymbol{M}=\begin{bmatrix} m-X_{\dot{u}} & 0 & 0 \\ 0 & m-Y_{\dot{v}} & 0 \\ 0 & 0 & I_z-N_{\dot{r}} \end{bmatrix}=\begin{bmatrix} m_{11} & 0 & 0 \\ 0 & m_{22} & 0 \\ 0 & 0 & m_{33} \end{bmatrix}$$

$$\boldsymbol{C}=\begin{bmatrix} 0 & 0 & -(m-Y_{\dot{v}})v \\ 0 & 0 & (m-X_{\dot{u}})u \\ (m-Y_{\dot{v}})v & -(m-X_{\dot{u}})u & 0 \end{bmatrix}=\begin{bmatrix} 0 & 0 & -m_{22}v \\ 0 & 0 & m_{11}u \\ m_{22}v & -m_{11}u & 0 \end{bmatrix} \tag{15}$$

$$\boldsymbol{D}=\begin{bmatrix} d_{11} & 0 & 0 \\ 0 & d_{22} & 0 \\ 0 & 0 & d_{33} \end{bmatrix},$$

where the parameters m_{11}, m_{22}, and m_{33} include the ship inertia including added mass effects, d_{11}, d_{22}, and d_{33} denote the damping-related coefficients, $X_{\dot{u}}$, $Y_{\dot{v}}$, and $N_{\dot{r}}$ are the hydrodynamic coefficients, and m and I_z denote the mass and rotational inertia of the underactuated marine vehicle, respectively.

Assumption 5: The body-fixed coordinate frame {b} (body frame) is located at a point $(x_P^*, 0)$, at a distance x_P^* from the vehicle's center of gravity along the center line of the ship.

Therefore, the 3-DOF model is expounded as:

$$\begin{aligned} \dot{x} &= u\cos\psi - v\sin\psi \\ \dot{y} &= u\sin\psi + v\cos\psi \\ \dot{\psi} &= r \\ \dot{u} &= \frac{m_{22}}{m_{11}}vr - \frac{d_{11}}{m_{11}}u + \frac{1}{m_{11}}\tau_u \\ \dot{v} &= -\frac{m_{11}}{m_{22}}ur - \frac{d_{22}}{m_{22}}v \\ \dot{r} &= \frac{m_{11}-m_{22}}{m_{33}}uv - \frac{d_{33}}{m_{33}}r + \frac{1}{m_{33}}\tau_r, \end{aligned} \tag{16}$$

where

$$\tau_u = T_s + T_p,\ \tau_r = \left(T_p - T_s\right)B/2. \tag{17}$$

T_p, T_s, and B refer to the control output of port propeller, starboard propeller, and beam length of the USV.

Considering the input saturation, $\boldsymbol{\tau} = [\tau_u, 0, \tau_r]^T$ denotes the actual control signal produced by the propellers, and $\boldsymbol{\tau}$ is written as

$$Sat_*(x) = \begin{cases} \tau_{*max}, & x > \tau_{*max} \\ x, & \tau_{*min} \le x \le \tau_{*max} \\ \tau_{*min}, & x < \tau_{*min}, \end{cases} \tag{18}$$

where $\tau_{*max}$and τ_{*min} denote the upper and lower bounds, respectively, where $*= u, r$. The desired control inputs should be x and the actual control inputs should be $\boldsymbol{\tau}$.

2.2.3. Problem description

Consider a global planner delivers the USV with a set of waypoint permutations or reference path. The USV should then properly navigate through the path that these waypoints have defined. Following a predetermined path without regard to time restrictions is referred to as path following [50]. An underactuated vessel could complete this mission with total velocity $U_d = \sqrt{u^2 + v^2}$ in the NED frame is tangential to the path. It is worth noted that the primary distinction between the trajectory tracking task and the path following task is that the path following task's path is elements that make up by a generic variable rather than time. This indicates that the vehicle is not necessary to arrive at a precise place along the curve at a particular time, but rather it must converge to the path and proceed through it at a constant speed.

To solve the aforementioned issue, a new reference frame is generated at the desired path $\gamma(s) = \{(x(s), y(s))|s \in \mathbb{R}\}$, where s is a scalar parameter. and travel along the curve with a constant speed $U > 0$. According to the definition, let's consider a virtual frame (VF) moves along $\gamma(s)$. For a waypoint $\boldsymbol{p}$ along the curve of the origin of VF, which we call $x(s)$, $y(s)$ is defined by the parameter s, and the path angle is $\psi(s)$. Now, objectives of the path following problem can be illustrated as:

$$\begin{aligned} &\lim_{t\to\infty} x_e = 0 \\ &\lim_{t\to\infty} y_e = 0 \\ &\lim_{t\to\infty} \psi_e = 0, \end{aligned} \tag{19}$$

where

$$\begin{bmatrix} x_e \\ y_e \\ \psi_e \end{bmatrix} = \begin{bmatrix} cos\ (\psi(s)) & -sin\ (\psi(s)) & 0 \\ sin\ (\psi(s)) & cos\ (\psi(s)) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x - x(s) \\ y - y(s) \\ \psi - \psi(s) \end{bmatrix}, \tag{20}$$

where x_e, y_e, and ψ_e represent the position and course angle error between the marine vehicle and the path.

3. Methodology

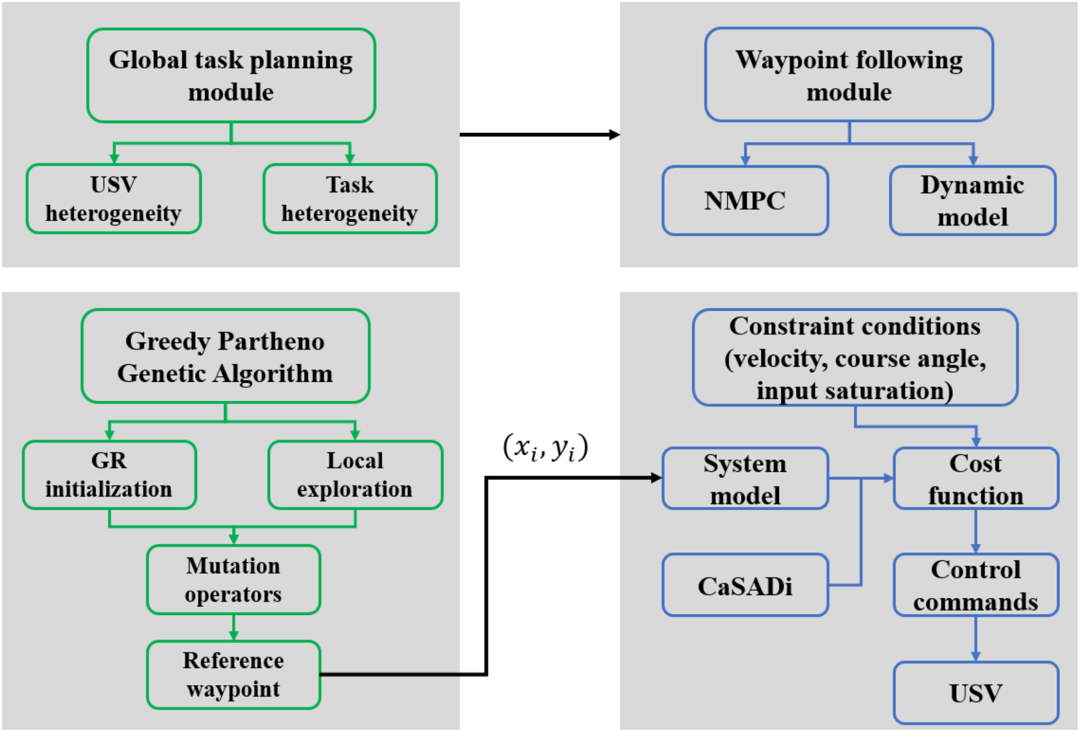

Fig. 3. Framework of the proposed method

The general framework of the methodology is illustrated in Fig. 3.

3.1.Greedy Partheno Genetic Algorithm

In this section, we propose the Greedy Partheno Genetic Algorithm (GPGA) and introduce how it solves the proposed EMTSP efficiently. Partheno genetic algorithm (PGA) is a modified version of GA that produces offspring through parthenogenesis. In lieu of conventional GA's mutation and crossover operators, PGA utilizes a series of operators on a single chromosome to produce offspring. Specifically, the crossover operator plays a crucial role in GA, whereas the mutation operator is typically regarded as an assisting operator. In PGA, however, the crossover operator is eliminated, and the mutation operator is considered the main operator. Consequently, PGA is more straightforward than GA in genetic operations, and initial population diversity is optional.

3.1.1. Dual-coded chromosome

The existing single chromosome and break-point type chromosome coding schemes are not suitable for EMTSP due to its heterogenous feature. To this end, we propose a dual-coded chromosome type that is decimally coded, i.e., task and USV chromosomes with the individual length being $N_T - 1$. The depot for all USVs is not coded in the chromosomes and is added to the final solution to meet the constraints. The first chromosome has a permutation of $N_T - 1$ tasks while the second assigns a USV to each of the common and exclusive tasks in the corresponding position of the first, following the task-USV matching relationship represented by Eq. (12).

A coding example of the chromosome with $N_T = 10$ and $N_U = 3$ is shown in Fig 4. Gene 1, 2, and 3 in the task chromosome are exclusive tasks for USV 1, gene 4 and 5 are exclusive for USV 2, gene 6 and 7 are exclusive for USV 3, respectively. It represents the task-USV matching relationship that must be met. The common tasks are genes 8-10 that can be accomplished by any USV. As denoted in the chromosome, task 2, 10, 1, and 3 (in that sequence) are visited by USV 1. Similarly, task 9, 5, and 4 (in that sequence) are visited by USV 2, and task 8, 7, and 6 are visited by USV 3.

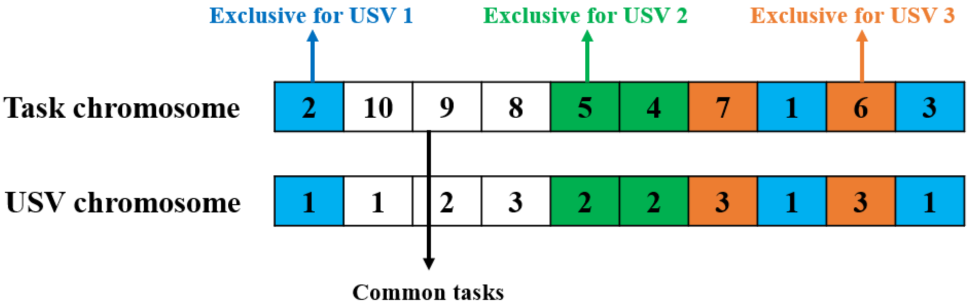

Fig. 4. Chromosome representation

3.1.2. Fitness function

With GPGA, the roulette selection is no longer used, and the fitness value is now calculated as the sum of the total distance and the difference between the maximum and minimum distances., see Eq. (21). The smaller the fitness function is, the better the quality of the individual is.

$$F = \sum_{k=1}^{N_U} D_k + (\underbrace{\text{Max}\, D_k}_{k=1,2,\ldots,N_U} - \underbrace{\text{Min}\, D_l}_{l=1,2,\ldots,N_U}). \tag{21}$$

3.1.3. Greedy randomized initialization

The GPGA begins its search with an initial population P of p high-quality solutions, often known as elite solutions. To develop an initial population, we generate a feasible solution using a greedy randomized heuristic. The initialization is illustrated by the following steps: 1) Use the exclusive task set to construct a subtour for each USV; 2) distribute the common tasks among the N_U subtours to get the solution. The pseudocode of the greedy randomized initialization is shown in Algorithm 1.

Algorithm 1. Greedy randomized initialization

Algorithm 1. Greedy randomized initialization

```
Input: Exclusive task sets $\{E_1, E_2, E_3, \ldots, E_{N_F}\}$, common task set $C$
Output: $p$        % feasible solution
```

```
3:  $p \leftarrow \emptyset$
4:  % Build $N_F$ partial routes with exclusive tasks
5:  for $k = 1{:}N_F$ do
6:      $r_k \leftarrow \{0\}$   % Initiate the route with task 0
7:      while $E_k \neq \emptyset$ do
8:          Randomly select task $i$ from $E_k$
9:          Insert task $i$ into $r_k$ with minimal distance increase
10:         Remove task $i$ from $E_k$
11:     end while
12:     $p \leftarrow p \cup \{r_k\}$
13: end for
14: % Dispatch the common tasks $C\backslash\{0\}$ among $N_F$ partial routes
15  $C' \leftarrow C\backslash\{0\}$
16: while $C' \neq \emptyset$ do
17:     Randomly select task $j$ from $C'$
18:     Insert task $j$ into route $p$ with total minimal distance increase
19:     Remove task $j$ from $C'$
20: end while
21: return $p$
```

Initiating the route with task 0 is the first step to create the k-th partial route r_k (lines 5-13). Next, uniformly selected exclusive tasks from E_k are introduced into r_k each one at a time, with the purpose of maximizing the route distance reduction. The first step terminates when each salesman's exclusive cities is entered into the corresponding route, yielding a partial solution p made up of N_F partial routes. The second phase (lines 15-20) involves uniformly processing the common tasks j from $C\backslash\{0\}$ and inserting them, one at a time, into a route of the partial solution p so as to minimize the increments of the distances of the solution.

3.1.4. Local exploration

When it comes to GPGA, local exploration is a crucial component that aids drive the discovery of solutions for quality improvement. GPGA uses a special method that clusters the tasks near to one other to examine local exploration and generate a better solution. The procedure is illustrated in Algorithm 2.

By clustering the tasks adjacent to each other, the exploration procedure can locally improve the solution optimality. P_A and P_B represent two randomly selected chromosome and the new chromosome is produced in various exploration directions with respect to the $flag$ value. Let us take P_A (see Fig. 5) as the parent, k as 3, and $flag$ as 1, the exploration starts from front to back, thereby result in the next nearby task to k, i.e., 6. On the contrary, if we take $flag$ as -1, the exploration starts from back to front in P_A, and the previous task adjacent to k is found, i.e., 9. The same applies to P_B. An example is shown in Fig. 5, whereby the task 1, 2, and 3 are exclusive for USV 1, 2, and 3, respectively. We take $k = 3$ as the start and $flag = 1$ (Suppose $flag$ will not change in this case).

Algorithm 2. Local exploration

Algorithm 2. Local exploration

```
Input: Two randomly selected chromosomes $P_A$, $P_B$
Output: new chromosome $P_C$
$L = P_A.length()$
$flag = rand\{-1, 1\}$
$k = rand(1, L)$, $k \in Z$
$P_C = [k]$
while $L > 1$ do
    if $flag == 1$ then
        $g_A = P_A.Adjacent(k, 1)$     % $Adjacent(k, 1)$: front to back
        $g_B = P_B.Adjacent(k,1)$
    **else**
        $g_A = P_A.Adjacent(k,-1)$      % $Adjacent(k,-1)$: back to front
        $g_B = P_B.Adjacent(k,-1)$
    **end if**
    $P_A.pop(k),\ P_B.pop(k),\ flag = rand\{-1,1\}$
    $d_A = \|k, g_A\|$
    $d_B = \|k, g_B\|$
    **if** $d_A < d_B$ **then** $k = g_A$
        **else** $k = g_B$
    **end if**
    $P_C.push(k)$
    $L = P_A.length()$
**end while**
**return** $P_C$
```

The procedure of the local exploration is illustrated as follows:

Step 1: Randomly choose two individuals P_A and P_B.

Step 2: Generate a chromosome based on Algorithm 2 and randomly choose a USV chromosome as the new USV chromosome.

Step 3: Check if each exclusive task is assigned to a correct USV and correct the wrong assignments if any.

Step 4: Rationalize the sequence and rank the genes by task order and category.

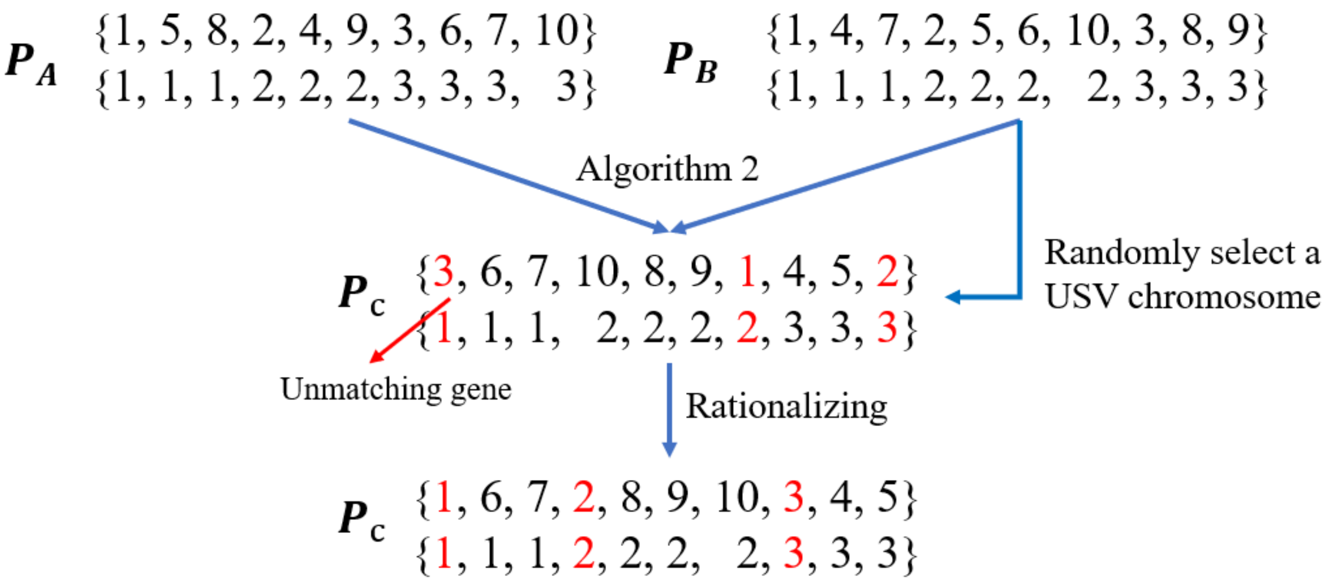

Fig. 5. Example of local exploration

3.1.5. *Mutation operators*

In order to prevent the search process getting trapped in a local optimum, the mutation operator is implemented. In this study, we develop four alternative mutation operators to increase the number of chromosomal variant forms and permutations available to the search process, hence increasing the likelihood of it to escape the local optimum. If a mutation is being selected in the children's chromosome, each of the four could be selected.

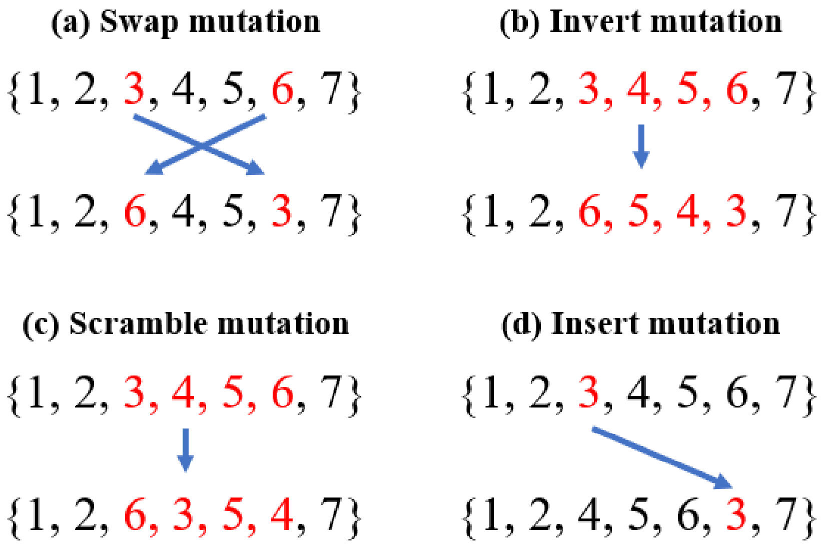

Fig. 6. Mutation operators

Swap mutation: Two tasks i and j, $i < j$, $i,j \in \mathfrak{S}$ on task chromosome are firstly randomly selected. Then, the two genes on the selected points are exchanged, as shown in Fig. 6 (a).

Invert mutation: Two tasks i and j, $i < j$, $i,j \in \mathfrak{S}$ on task chromosome are firstly randomly selected as the subtour. Then, the order of the subtour between i and j is inverted, see Fig. 6 (b).

Scramble mutation: Two tasks i and j, $i < j$, $i,j \in \mathfrak{S}$ on task chromosome are firstly randomly selected as the subtour. Then, the order of the subtour between i and j is scrambled, see Fig. 6 (c).

Insert mutation: One task i, $i \in \mathfrak{S}$ on task chromosome is randomly selected, removes it from the chromosome, and inserts it in a randomly selected place, see Fig. 6 (d).

The process of mutation of the GPGA are described as follows:
Step 1: Select five members randomly from the current population who have not already been chosen before.
Step 2: Find the one has the best fitness value in the 5 members.
Step 3: Generating a temporary population that consists of 5 members. Each of the individual is assigned to the value of the individual selected in step 2.
Step 4: Generate 2 random points i and j, or the insertion location i.
Step 5: Mutate each individual in the test group created in step 3 in the following procedures:

(1) The first one will stay the same.
(2) The second one will mutate by swapping.
(3) The third one will mutate by inverting.
(4) The fourth one will mutate by scrambling.
(5) The last one will mutate by inserting.
(6) Check if each exclusive task is assigned to a correct USV and correct the wrong assignments if any.

Step 6: Merge the generated population into the original.
Step 7: If all individuals in the current population have been selected, then move on; otherwise, return to step 1.

Swap, Invert, Scramble, and Insert are all PGA mutation procedures that find their way into GPGA as well. Therefore, GPGA shares the same benefits as PGA. Additionally, step 5 ensures that the best individual identified in step 2 will be passed on to the next generation. New individuals will be generated by altering the finest possible one in several ways. Some of the new individuals are produced by changing the route of the best individual. Also, a few are made by adjusting the best individuals' USV-task matching relationships. Others are produced by altering both produces. In this way, the mutation operation takes into account both the route sequence and the matching relationship. Additionally, since the selected individual to be altered to generate children is a relatively decent individual, step 5 ensures that the algorithm will recognize the second-best options in the iterative process. The foregoing suggests that GPGA outperforms over PGA in both global search capability and local escaping ability. Generally speaking, GPGA makes it simpler to arrive at the optimal potential solution.

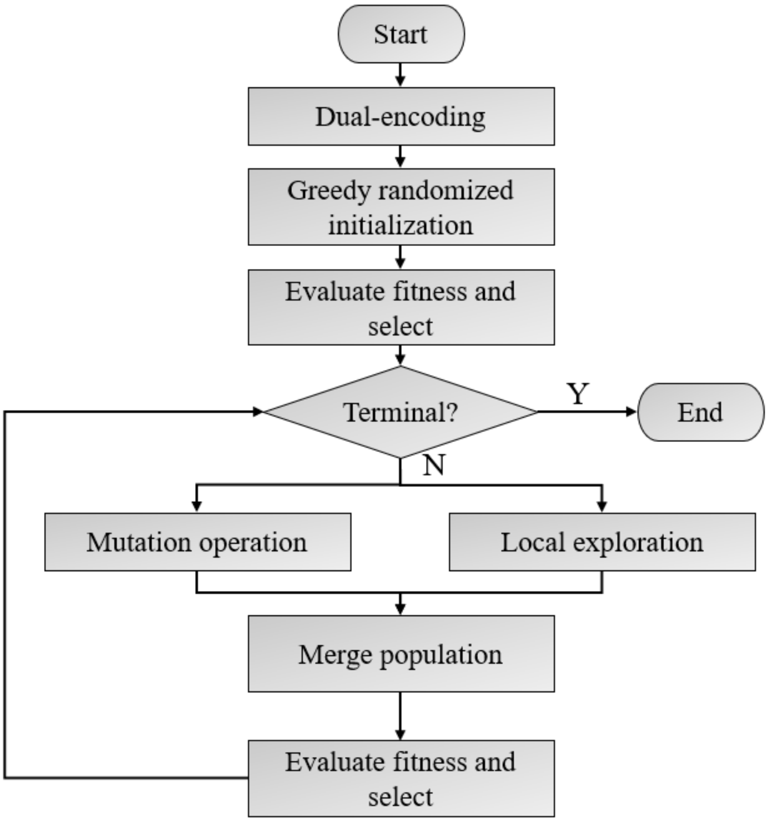
Start
Dual-encoding
Greedy randomized initialization
Evaluate fitness and select
Terminal?
Y
End
N
Mutation operation
Local exploration
Merge population
Evaluate fitness and select

Fig. 7. Flowchart of GPGA

3.1.6. Algorithm flow

For an objective function $F(X)$, GPGA will find an X^* such that $\forall X,\ F(X^*) < F(X)$. The proposed GPGA are illustrated as following, see Fig. 7:

(1) Generate the initial population using the greedy randomized initialization.

(2) Evaluate the fitness value of each individual X_i^g (g is the number of current generations, i is the index for individual) in the initial set. The scheme of elitism dictates that the one with the highest fitness will be replicated and labeled as X^*.

(3) Implement mutation and local exploration using the strategies depicted in Section 3.1.4 and Section 3.1.5 to produce offspring.

(4) Evaluate the fitness value of X_i^{g+1} in the new generation. If $F\left(X_i^{g+1}\right) \leq F(X^*)$, X_i^{g+1} is replicated and labeled as X^*.

(5) If the terminal condition holds, the best answer, X^*, should be exported or set $g = g + 1$, then turn to (3).

3.2.Nonlinear model predictive control

3.2.1. State space model

Tracking error (x_e, y_e) minimization is the primary goal of NMPC control during path-following. In addition, it is preferred that the USV's course angle corresponds to that of the path angle, guaranteeing that ψ_e

converge to zero. Hence, combining the error dynamics and USV dynamics, we introduce the following state space model:

$$\begin{aligned}
\dot{x}_e &= u\cos(\psi - \psi(s)) - v\sin(\psi - \psi(s)) \\
\dot{y}_e &= u\sin(\psi - \psi(s)) + v\cos(\psi - \psi(s)) \\
\dot{\psi}_e &= r \\
\dot{x} &= u\cos\psi - v\sin\psi \\
\dot{y} &= u\sin\psi + v\cos\psi \\
\dot{u} &= \frac{m_{22}}{m_{11}}vr - \frac{d_{11}}{m_{11}}u + d_1 \\
\dot{v} &= -\frac{m_{11}}{m_{22}}ur - \frac{d_{22}}{m_{22}}v + d_2 \\
\dot{r} &= \frac{m_{11} - m_{22}}{m_{33}}uv - \frac{d_{33}}{m_{33}}r + d_3.
\end{aligned} \tag{22}$$

Therefore, the vessel model is written in a compact form as:

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}) + g_1(\boldsymbol{x})\boldsymbol{u} + g_2(\boldsymbol{x})\boldsymbol{w}, \tag{23}$$

where $\boldsymbol{x} = [x_e, y_e, \psi_e, x, y, u, v, r]^T$ is the state vector, $\boldsymbol{u} = [\tau_u,\ \tau_r]^T$ is the input vector of surge force and yaw moment, $\boldsymbol{w} = [d_1, d_2, d_3]^T$ is the disturbance on surge, sway, and yaw, $g_1(\boldsymbol{x})$ and $g_2(\boldsymbol{x})$ are the control and disturbance configuration matrices, respectively, with the following structure:

$$\begin{gathered}
g_1(\boldsymbol{x}) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & \dfrac{1}{m_{11}} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \dfrac{1}{m_{33}} \end{bmatrix}^T \\
g_2(\boldsymbol{x}) = \begin{bmatrix} 0_{5\times 3} \\ M^{-1}R(\psi) \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & \dfrac{\cos\psi}{m_{11}} & \dfrac{\sin\psi}{m_{22}} & 0 \\ 0 & 0 & 0 & 0 & 0 & \dfrac{\sin\psi}{m_{11}} & \dfrac{\cos\psi}{m_{22}} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \dfrac{1}{m_{33}} \end{bmatrix}^T,
\end{gathered} \tag{24}$$

As a result of the lack of a sway control force created by the actuators, the controller is unable to reject the disturbance in the sway direction for the undereducated configuration. By the definitions illustrated in [40], the heading angle ψ in the state vector should be replaced by the course angle χ, since in the presence of an external force in the sway direction, the vessel will not be able to attain zero tracking error for the heading angle. A nonzero sideslip angle will result regardless of the heading angle, but the resulting force component will counteract the sway disturbance that would have happened had the sideslip angle been zero. Based on that, the state vector is rewritten by

$$\boldsymbol{x} = [x_e, y_e, \chi_e, x, y, u, v, r]^T, \tag{25}$$

where $\chi_e = \chi - \chi_r$ is the tracking error which considers the sideslip angle.

3.2.2. NMPC design

By discretizing the continuous-time model in Eq. (23), we obtain the dynamic system under the control of the proposed NMPC. As a result, we may express the desired control system's discrete-time model as:

$$\boldsymbol{x}(k+1) = f\big(\boldsymbol{x}(k), \boldsymbol{u}(k), \boldsymbol{w}(k)\big), \tag{26}$$

Here, the state $\boldsymbol{x}$ is comprised by the error and vessel dynamics, and the input $\boldsymbol{u}$ is the input vector.

Moreover, in contrast to existing LOS-based guidance strategies, the NMPC framework presented in this paper is able to consider physical constraints of the mechanical system. We set low-level controller limitations on both velocity and the rate of change in surge and heading angle. Hence, the system should satisfy:

$$\begin{aligned} \boldsymbol{u}(k) \in \mathcal{U} = \left\{ \begin{bmatrix} 0 \\ -\pi \end{bmatrix} \leq \boldsymbol{u} < \begin{bmatrix} u_{max} \\ \pi \end{bmatrix} \right\} \\ \Delta\boldsymbol{u} = \boldsymbol{u}(k) - \boldsymbol{u}(k-1) \in \mathcal{U}_g = \left\{ \Delta\boldsymbol{u} \leq \begin{bmatrix} \tau_{u\_max} \\ \tau_{r\_max} \end{bmatrix} \right\}. \end{aligned} \tag{27}$$

Thus, nonlinear model predictive control (NMPC) is interpreted as the online acquisition of state feedback $\boldsymbol{u}(k)$ via a least-squares (LS) optimum control problem, whereby the objective function penalizes the amount by which the system's inputs and states deviate from their reference paths. It is expressed as:

$$\min_{x(k), \boldsymbol{u}(k)} \boldsymbol{J}_N(\boldsymbol{x}, \boldsymbol{u}) = \sum_{k=0}^{N-1} \ell(\boldsymbol{x}(k), \boldsymbol{u}(k)) + F\big(\boldsymbol{x}(N)\big), \tag{28}$$

s.t.

$$\begin{aligned} & \boldsymbol{x}(k+1) = f(\boldsymbol{x}(k), \boldsymbol{u}(k), \boldsymbol{w}(k)) \\ & \boldsymbol{u}(k) \in \mathcal{U} \\ & \Delta\boldsymbol{u} \in \mathcal{U}_g \\ & T_{p_{min}} \leq T_p \leq T_{p_{max}} \\ & T_{s_{min}} \leq T_s \leq T_{s_{max}} \\ & \ell\big(\boldsymbol{x}(k), \boldsymbol{u}(k)\big) > 0, \forall \boldsymbol{x}(k), \boldsymbol{u}(k), \end{aligned} \tag{29}$$

where $\ell(\boldsymbol{x}(k), \boldsymbol{u}(k))$ is the stage cost function and $F(\boldsymbol{x}(N))$ is the terminal cost function, $N > 0$ is the length of both the prediction and control horizons. The stage and terminal cost are defined as:

$$\ell\big(\boldsymbol{x}(k), \boldsymbol{u}(k)\big) = \|\boldsymbol{x}(k) - \boldsymbol{x_r}(k)\|_{\boldsymbol{Q}} + \|\boldsymbol{u}(k)\|_{\boldsymbol{R}} \tag{30}$$

$$F\big(\boldsymbol{x}(N)\big) = \|\boldsymbol{x}(N) - \boldsymbol{x_r}(N)\|_{\boldsymbol{P}}. \tag{31}$$

Here $\boldsymbol{J}_N$ is designed cost function, consists of the stage cost ℓ and the terminal cost F, the predictive horizon is denoted as N, $\boldsymbol{x_r}(k)$ and $\boldsymbol{x_r}(N)$ are the reference states of the path and predicted reference states, $\boldsymbol{Q}$, $\boldsymbol{R}$, and $\boldsymbol{P}$ are positive semidefinite weighing matrices. Control actions are penalized in order to discourage the application of high-energy, which could cause the system to be unstable.

3.2.3. Solver

This paper uses the CasADi software to solve the NMPC problem (28) subject to the restrictions given by (29). CasADi is a C++ program that can model and solve optimization problems with a great deal of flexibility, all while generating extremely efficient C++ code for real-time implementation and MATLAB executable (mex) files, used for simulation with MATLAB. It finds widespread use in fields like industrial control and robotics. In particular, nonlinear programming (NLP) solvers take a shooting-based approach to dynamic optimization. We use the direct single-shooting method since our investigations showed that the solution speed of the direct single-shooting approach is greater than that of the multi-shooting method when the number of prediction horizon steps is less than 30.

The solving process is illustrated as follows:

Step 1: Set number of sampling instants in the time prediction horizon N, and the sampling time T.
Step 2: Set weight matrices $\boldsymbol{Q}$, $\boldsymbol{R}$, $\boldsymbol{P}$.
Step 3: Set model constraints.
Step 4: Set state references $\boldsymbol{x_r}(k)$, $\boldsymbol{x_r}(N)$**.**
Step 5: Get the current value $\boldsymbol{x}$**.**
Step 6: Solve the NLP problem Eq. (28), and take the optimal control input vector $[\boldsymbol{u}_0, \boldsymbol{u}_1, \boldsymbol{u}_2, \dots, \boldsymbol{u}_{N-1}]$ and the predicted states $[\boldsymbol{x}_0, \boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_{N-1}]$.
Step 7: Apply the first element $\boldsymbol{u}_0$, and go to **Step 5**.

3.2.4. Stability

Despite numerous studies on the asymptotic stability of the NMPC problem, creating acceptable conditions remains an unresolved challenge that may make online optimizations more complex and time expensive to accomplish. Hence, we only discuss the stability concerning our model. According to a series of studies in [52–54], stability can be ensured for finite horizon problems under several conditions. They are presented as following:

1. $\boldsymbol{u}$ is compact, and $\boldsymbol{x}$ is connected and contains the origin in the interior of $\boldsymbol{u} \times \boldsymbol{x}$**.**
2. $\exists \boldsymbol{u} \in \mathcal{U}$ which makes $f(\boldsymbol{x_r}, \boldsymbol{u}) = \boldsymbol{x_r}$.
3. Objective function $\boldsymbol{J}$ should satisfy $\boldsymbol{J}(\boldsymbol{x_r}, \boldsymbol{u}) = 0$, from $\boldsymbol{u} \in \mathcal{U}$ obtained from the second assumption.

Because there are no further limitations on the states in our problem, we may assume that the feasible set always has the origin in the interior via simple axis transformation. Typically, linear inequalities are chosen as control constraints, so $\boldsymbol{u}$ is a compact set. For the second assumption, it is easily checked by observing the USV system. The third assumption will be satisfied as long as the cost function $\boldsymbol{J}$ is the quadratic, as Eq. (30).

4. Results and discussion

To evaluate the proposed strategy, several illustrative simulations are conducted progressively by global path planning, waypoint following, and the combination of the two modules. They are performed via MATLAB R2021a environment with a PC that is configured with a 2.10-GHz Intel(R) Core (TM) i7-1260P processor and 16.0-GB RAM.

4.1.Simulation: global task planning

4.1.1. Convergence test

In this subsection, simulation studies and comprehensive comparisons are provided to validate the convergence characteristic and solution quality of GPGA in solving the global path planning problem. In order to facilitate simulations, we conduct the performance evaluation using classical instances from TSPLIB, see Table 1. To show the improvement effect of the novel strategies, methods from existing references, including IPGA [55] and MOGA [56], are applied to solve the problem. For fair comparison, we run 30 times on each instance and perform statistical analysis regarding time cost and solution optimality. It is worth to note that since we only need to test the convergence characteristic, heterogeneity is not considered. Therefore, the process of checking the USV-task matching relationship is skipped.
The parameters are set as follows, the population number and iteration are set as the same while other parameters are set by the best value according to [55,56]:

- GPGA: $M = 100$, $T_m = 4000$, mutation = 0.3, $P_{le} = 0.7$ (probability of local exploration).
- IPGA: $M = 100$, $T_m = 4000$, mutation = 0.01.
- MOGA: $M = 100$, $T_m = 4000$, mutation = 0.3, the crossover probability is adaptive.

Table 1 Setting of test instances

Instance	Number of tasks	Number of salesmen

Eil51	51	3
Berlin52	52	4
Eil76	76	4
Pr76	76	5
Rat99	99	6
Rat99	99	7

Table 2 Comparison of time cost

Instance	GPGA	IPGA	MOGA
Eil51(3)	7.33	8.62	10.79
Berlin52(4)	7.47	8.69	10.63
Eil76(4)	8.21	9.74	10.55
Pr76(5)	8.10	9.52	11.07
Rat99(6)	10.23	12.21	13.79
Rat99(7)	10.17	12.45	13.92

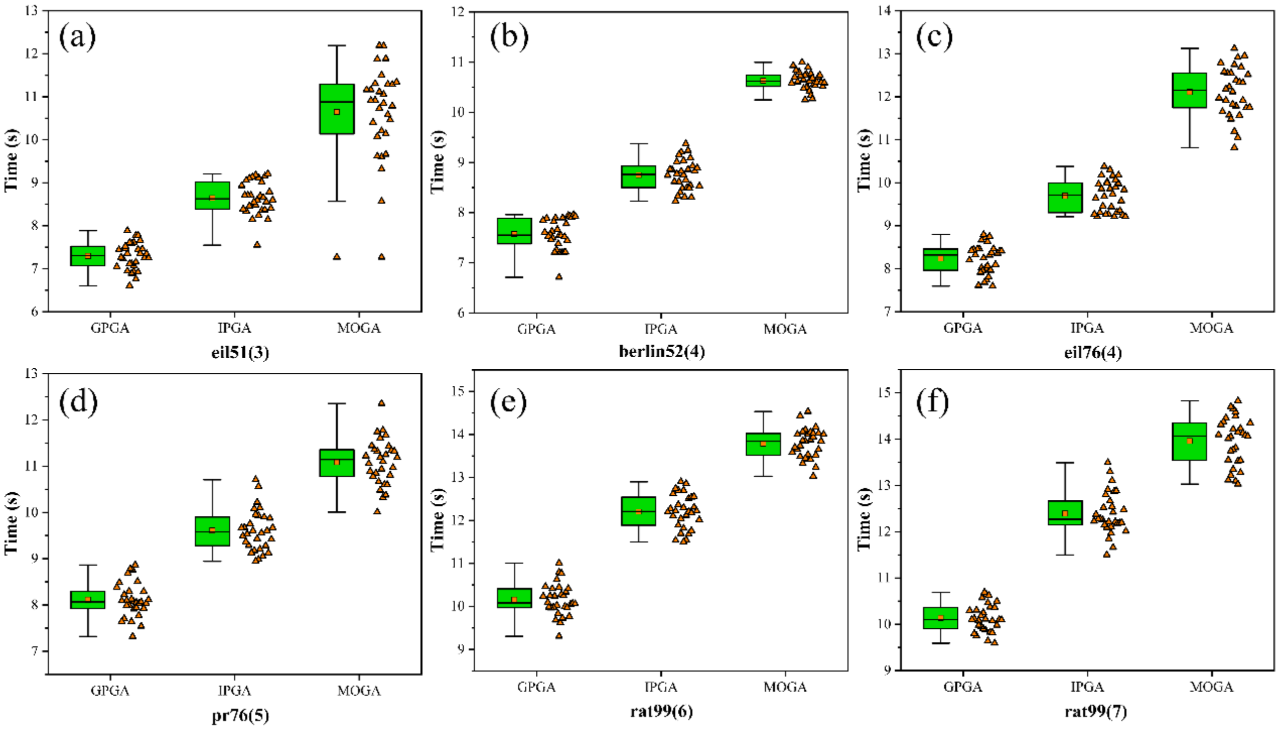

Fig. 8. Box-whisker plot of time cost

Computational results of GPGA and reference algorithms on the convergence efficiency are shown in Fig. 8 and Table 2. As denoted in Table 2, GPGA can solve the general MTSPs more quickly than the existing algorithms. With approximately 10% and 20% lower time cost compared to IPGA and MOGA, GPGA has shown its low computational burden in the algorithm process. Compared to the MOGA, GPGA and IPGA are generally more computational efficiency because the complex crossover procedure and roulette wheel selection are not performed, which significantly decreases the complexity. Moreover, compared to IPGA, our algorithm is superior since the mutation operation is much simpler but remains a high-level searching ability. As shown in Fig. 8, the IQR (range of the box) of GPGA is smaller than the reference algorithms in most cases (except for berlin52). The results indicate that the proposed method can achieve satisfactory computational stability results.

Table 3 Comparison of the solution quality

	GPGA		IPGA		MOGA	
	Avg (m)	468.87	Avg (m)	520.36	Avg (m)	500.07
Eil51	Best (m)	**446.37**	Best (m)	462.95	Best (m)	452.96
	SD (m)	16.04	SD (m)	27.73	SD (m)	29.11
Berlin52	Avg (m)	8778.31	Avg (m)	9811.12	Avg (m)	9295.84

	Best (m)	**7874.43**	Best (m)	8609.51	Best (m)	8447.83
	SD (m)	583.04	SD (m)	615.22	SD (m)	505.17
	Avg (m)	669.06	Avg (m)	806.02	Avg (m)	738.17
Eil76	Best (m)	**625.37**	Best (m)	717.20	Best (m)	642.34
	SD (m)	29.00	SD (m)	38.48	SD (m)	42.62
	Avg (m)	131362	Avg (m)	140870	Avg (m)	137452
Pr76	Best (m)	**118854**	Best (m)	126273	Best (m)	123974
	SD (m)	8293.41	SD (m)	7142.23	SD (m)	7093.82
	Avg (m)	1399.21	Avg (m)	1579.21	Avg (m)	1524.20
Rat99	Best (m)	**1292.83**	Best (m)	1398.24	Best (m)	1367.24
	SD (m)	48.27	SD (m)	64.38	SD (m)	63.66
	Avg (m)	1445.21	Avg (m)	1604.12	Avg (m)	1553.50
Rat99	Best (m)	**1344.34**	Best (m)	1510.63	Best (m)	1408.62
	SD (m)	51.54	SD (m)	51.51	SD (m)	64.31

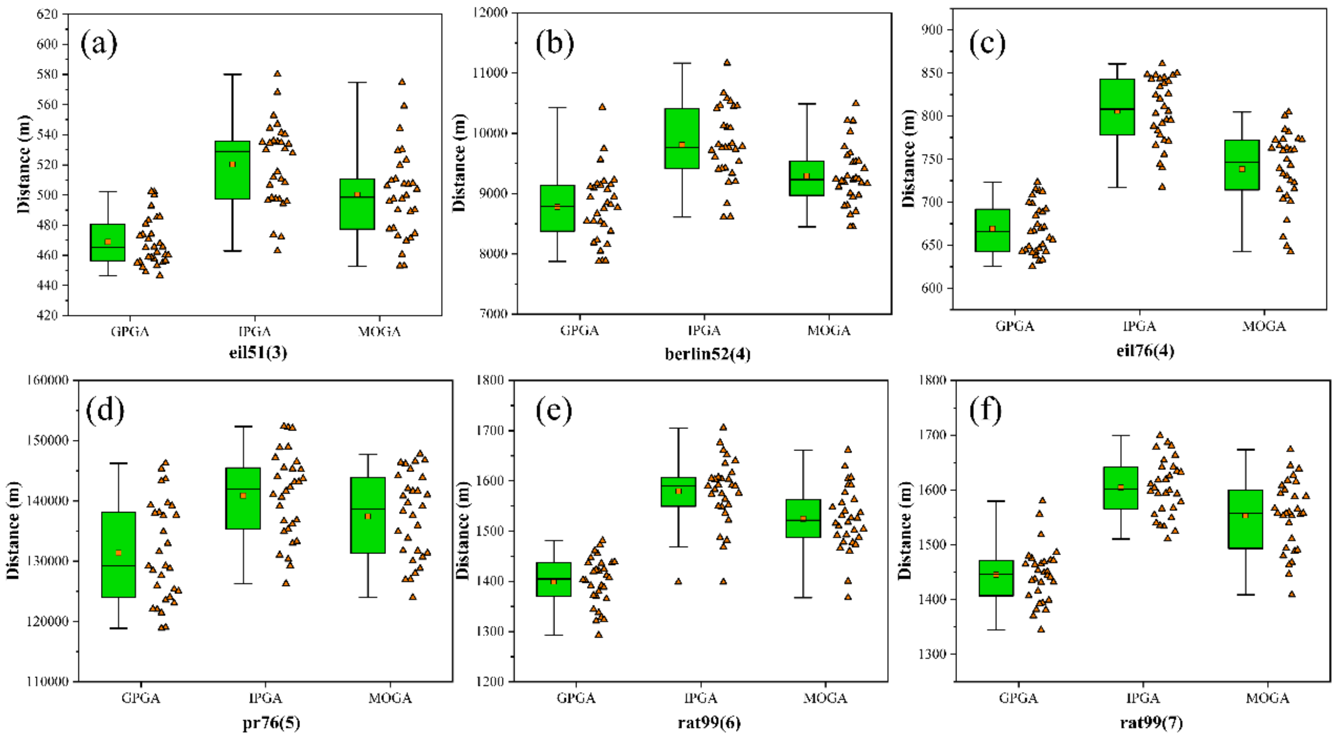

Fig. 9. Box-whisker plot of total distance

Table 3 presents the results of the compared algorithms on the six instances in terms of their solution optimality. As denoted by the best value (in bold), GPGA can optimally solve the MTSPs in the simulations. Compared to IPGA and MOGA, the proposed method yielded better results on the average and minimum distance. This indicates that GPGA merits strong global searching ability while preventing the local optimum effectively. This is contributed by the local exploration and mutation strategy. On the one hand, local exploration generates better offspring than the parents by gathering the nearby tasks, which improves the population quality. On the other hand, the various mutation strategies can help the algorithm jump out of the local optimum, thereby contributing to the global searching ability. In terms of stability, GPGA has also shown better results in most cases (except for pr76), see Fig. 9. Note that the results of IPGA are somewhat inconsistent (eil51 and pr76, the bound is large). We assume it is because the algorithm has fallen into numerous local optimums and yielded diverse results.

4.1.2. Heterogeneous task planning

In this subsection, to demonstrate the effects of the heterogeneity of USVs, four cases of experiments for the comprehensive analysis are designed. All four EMTSPs are assigned in turn with task sizes 40, 50, 60, and 70, and the common tasks and exclusive tasks are grouped according to Table. 4. It is worth to note that all the tasks are randomly distributed in the 2-D workspace (100*100 m) where the simulations are carried out. Each

USV departs from its base station and returns after completing the assigned tasks. The parameter setting is the same as in Section 4.1.1.

The four EMTSPs are depicted in Fig. 10. As denoted in the figure, the common tasks, exclusive tasks for USV1, exclusive tasks for USV2, and exclusive tasks for USV3 and USV4 are marked with black circles, blue triangles, magenta pentagram, red squares, and green pentagram, respectively.

Table 4 Design of EMTSPs

Case	Tasks count	USV count	Common tasks	Exclusive tasks
1	40	3	25	5 for each
2	50	4	30	5 for each
3	60	3	39	7 for each
4	60	4	40	5 for each

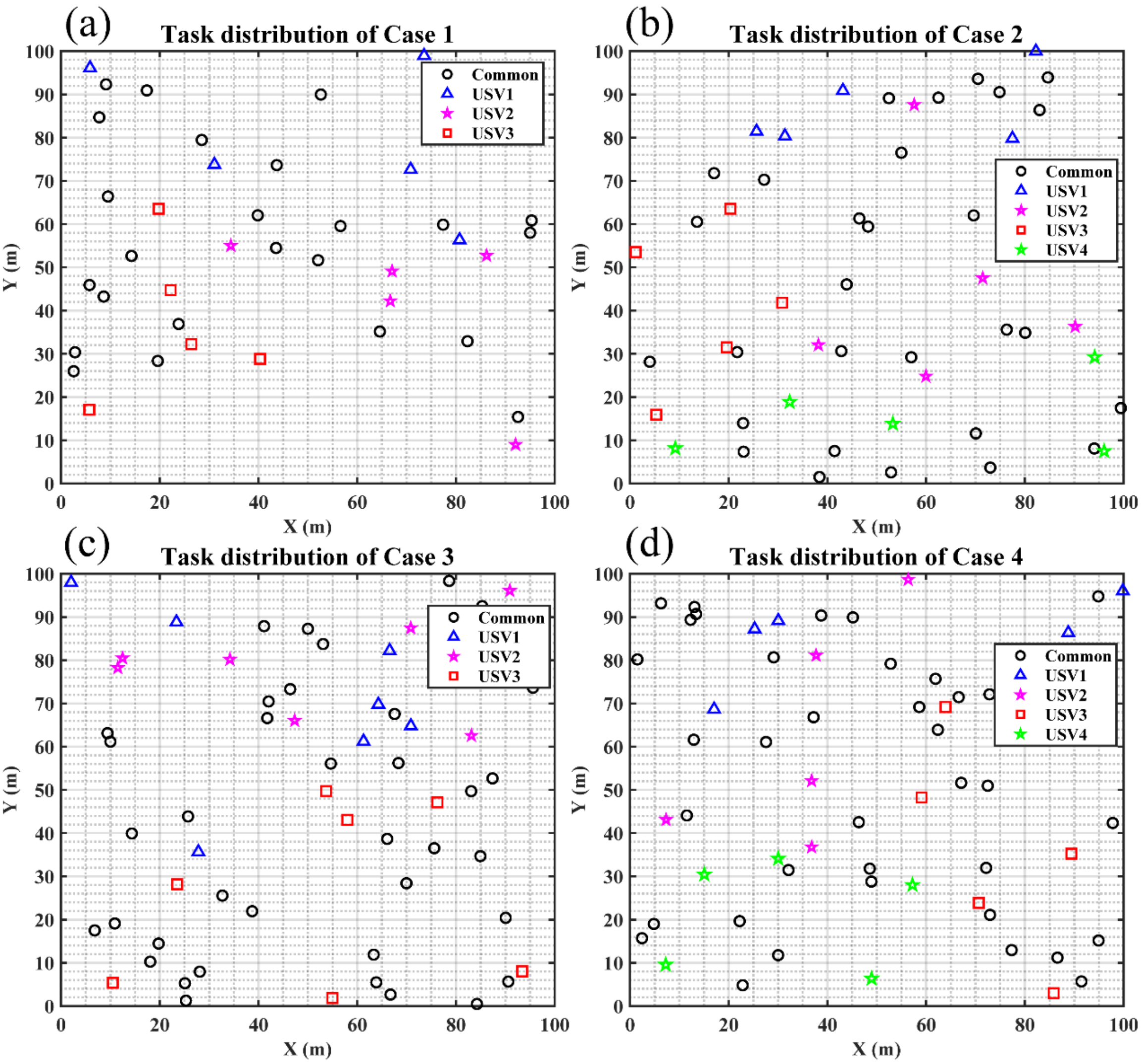

Fig. 10. Task distribution (a) Case 1; (b) Case 2; (c) Case 3; (d) Case 4

The convergence history of the four cases is shown in Fig. 11. As denoted by the convergence curve and time cost measurements, the computational cost slightly increased compared to the previous results. This is caused by the checking and correction procedure. Nevertheless, the proposed algorithm can still find the optimal solution without sacrificing computational efficiency. As indicated by Fig. 12, all the USVs with exclusive functional types have successfully completed their corresponding tasks. The checking and correcting of the USV-task matching relationship are performed after each genetic operation is completed, thereby ensuring no violation of the matching requirements. This indicates that our proposed model can perfectly handle heterogeneous path planning.

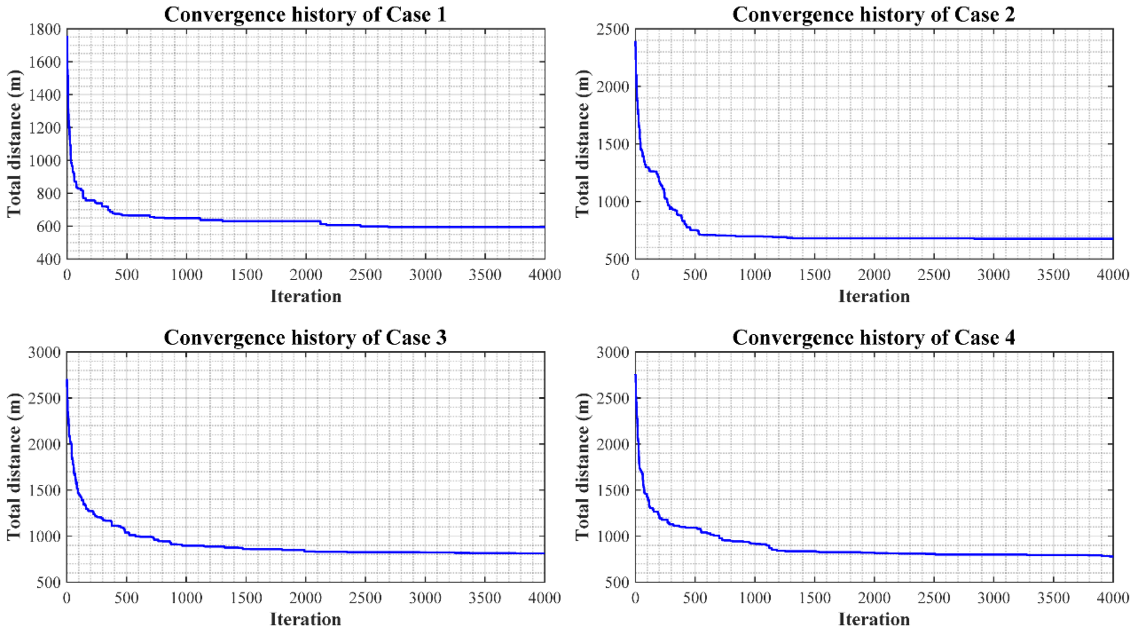

Fig. 11. Convergence history: the time costs for Case 1-4 are 8.82 s, 9.77 s, 11.02 s, and 11.34 s, respectively

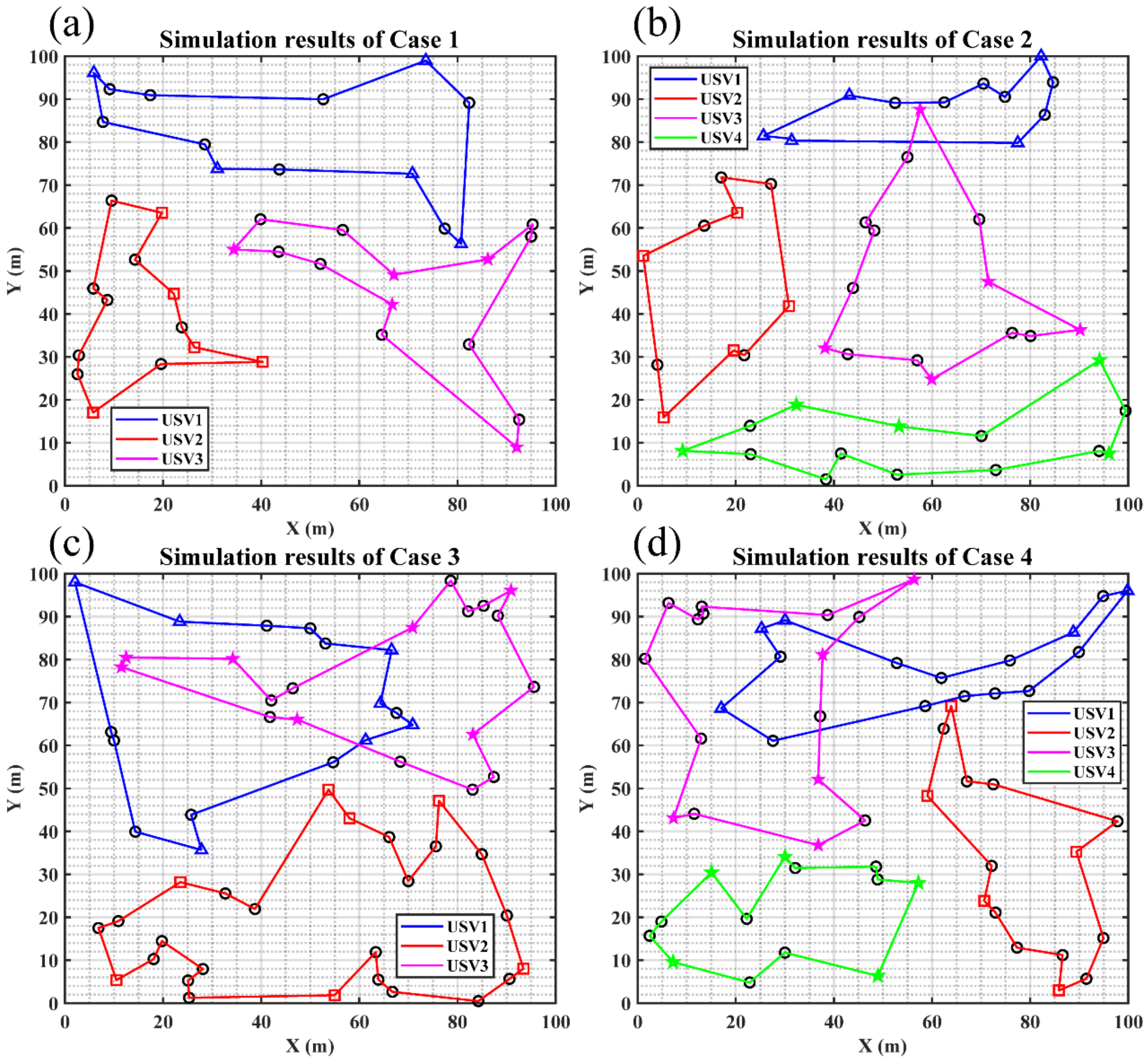

Fig. 12. Planning results

4.2.Simulation: waypoint following

In this section, simulation results are presented to demonstrate the validity and assess the performance of the proposed NMPC for waypoint tracking of an unmanned surface vehicle. In order to show the advantage of the efficiency of our model, comparative studies are also conducted with the well-known path following method Integral line of sight (ILOS) [41] and the adaptive LOS (ALOS) guidance [40]. The three methods are applied to the Otter surface vehicle, see Fig. 13. The Otter unmanned surface vehicle is a 2-[m]-long and 1.08-[m]-wide robotic platform developed by MARINE ROBOTICS (www.marinerobotics.com). The particulars and mechanical properties are shown in Table 5 and Table 6, respectively, and for more detailed information on the USV model, the readers are referred to the MSS toolbox (https://github.com/cybergalactic/MSS).

Table 5 Parameters of the Otter

Parameters	Explanations	Values	Units
M	Mass	65	kg
L	Length	2	m
B	Beam	1.08	m
N_p	number of propellers	2	-

Table 6 Maneuvering derivatives of the USV model

Inertial related	Value	Damping related	Value
m_{11}	85.28	d_{11}	-77.55
m_{22}	162.50	d_{22}	-0.02
m_{33}	41.45	d_{33}	-41.45

Fig. 13. USV Otter

The parameters settings are expounded and explained here. The prediction horizon length is selected to be 30 s, the control horizon is set as 2 s, and the sampling time is 0.1 s. $u_{min} = -3\ m/s,\ u_{max} = 3\ m/s,\ r_{min} = -0.1\ rad/s,\ r_{max} = 0.1\ rad/s,\ T_{p_{max}} = T_{s_{max}} = 119.7\ N,\ T_{p_{min}} = T_{s_{min}} = -66.7\ N,\ u_d = 1.9\ m/s$. The weight matrices were selected based on a series of simulation tests: $\boldsymbol{Q} = diag([0.5, 0.5, 2, 0, 0, 30, 0, 0.1])$, $\boldsymbol{P} = diag([1, 1, 4, 0, 0, 30, 0, 0.2])$, $\boldsymbol{R} = diag([0.001, 0.001])$. The weights on matrix $\boldsymbol{Q}$ and $\boldsymbol{P}$ were meant to penalize deviation from the path and course angle, and failure to maintain the required speed. Similarly, the weights on $\boldsymbol{R}$ penalizes aggressive changes in the control signals, to achieve a smooth thruster signal.

As denoted in Section 3.2, the disturbances affecting the system can be due to different external sources, such as wind, waves, and currents. In order to study the performance of NMPC under specific environment disturbances, time-varying environmental disturbances are employed on surge (d_1), sway (d_2), and yaw components (d_3), see the following equation and Fig. 14:

$$\begin{aligned} d_1 &= 1.5\sin t + \cos 0.7t \\ d_2 &= 1.7\sin 1.3t + 0.8\cos 0.8t \\ d_3 &= 1.2\sin 0.5t + 1.3\cos 1.2t. \end{aligned} \tag{26}$$

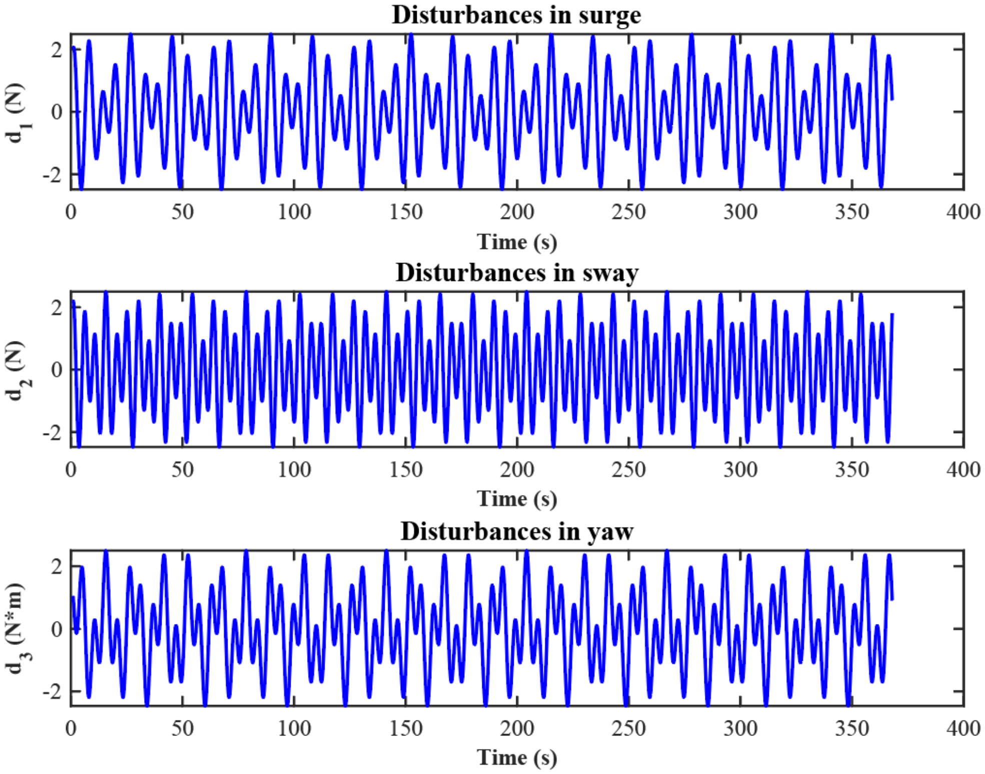

Fig. 14. Tracking results

4.2.1. *Test 1: Simulation under different model uncertainties*

In this subsection, we will test the robustness of the NMPC with respect to the different level of model uncertainties. The test cases are threefold: (1) nominal model without disturbances and model uncertainty; (2) model with disturbances and 10% model uncertainty; (3) model with disturbances and 20% model uncertainty. The model uncertainties of the Otter vehicle are $\boldsymbol{\varDelta M}$ and $\boldsymbol{\varDelta D}$, which is randomly generated at each time within the uncertainty boundary. Therefore, the inertial matrix value and damping matrix value would vary according to the following equation at each time step:

$$\begin{aligned} \boldsymbol{M} &= \boldsymbol{M} \pm \boldsymbol{\varDelta M} \\ \boldsymbol{D} &= \boldsymbol{D} \pm \boldsymbol{\varDelta D}. \end{aligned} \tag{27}$$

In this simulation, we assume the boundary of the model uncertainties are $\boldsymbol{\varDelta M} \in (0, 0.1\boldsymbol{M})$ and $\boldsymbol{\varDelta D} \in (0, 0.1\boldsymbol{D})$ for the 10% case, while $\boldsymbol{M} \in (0, 0.2\boldsymbol{M})$ and $\boldsymbol{\varDelta D} \in (0, 0.2\boldsymbol{D})$ for the 20% case. The reference path is designed as:

$$\begin{aligned} x_d &= 6.25\omega + 50\sin(2\pi\omega/40) \\ y_d &= 8.75\omega - 0.05\omega^2, \end{aligned} \tag{28}$$

where ω is the path parameter that is independent of time. The USV is originally positioned at $x = 0\,m$, $y = 0\,m$, while the initial point of the path is assumed to be $\omega = 0$ and it ends at $\omega = 50$.

To analyze the results in more details, IAE (integrated absolute errors) is employed to compare the steady-state and the transient response performance quantitatively. IAE of longitudinal and lateral position can be defined as:

$$IAE_x = \int_0^t |x_e(\omega)| d\omega$$
$$IAE_y = \int_0^t |y_e(\omega)| d\omega, \tag{29}$$

where x_e and y_e is the tracking error in the longitude and lateral direction, respectively.

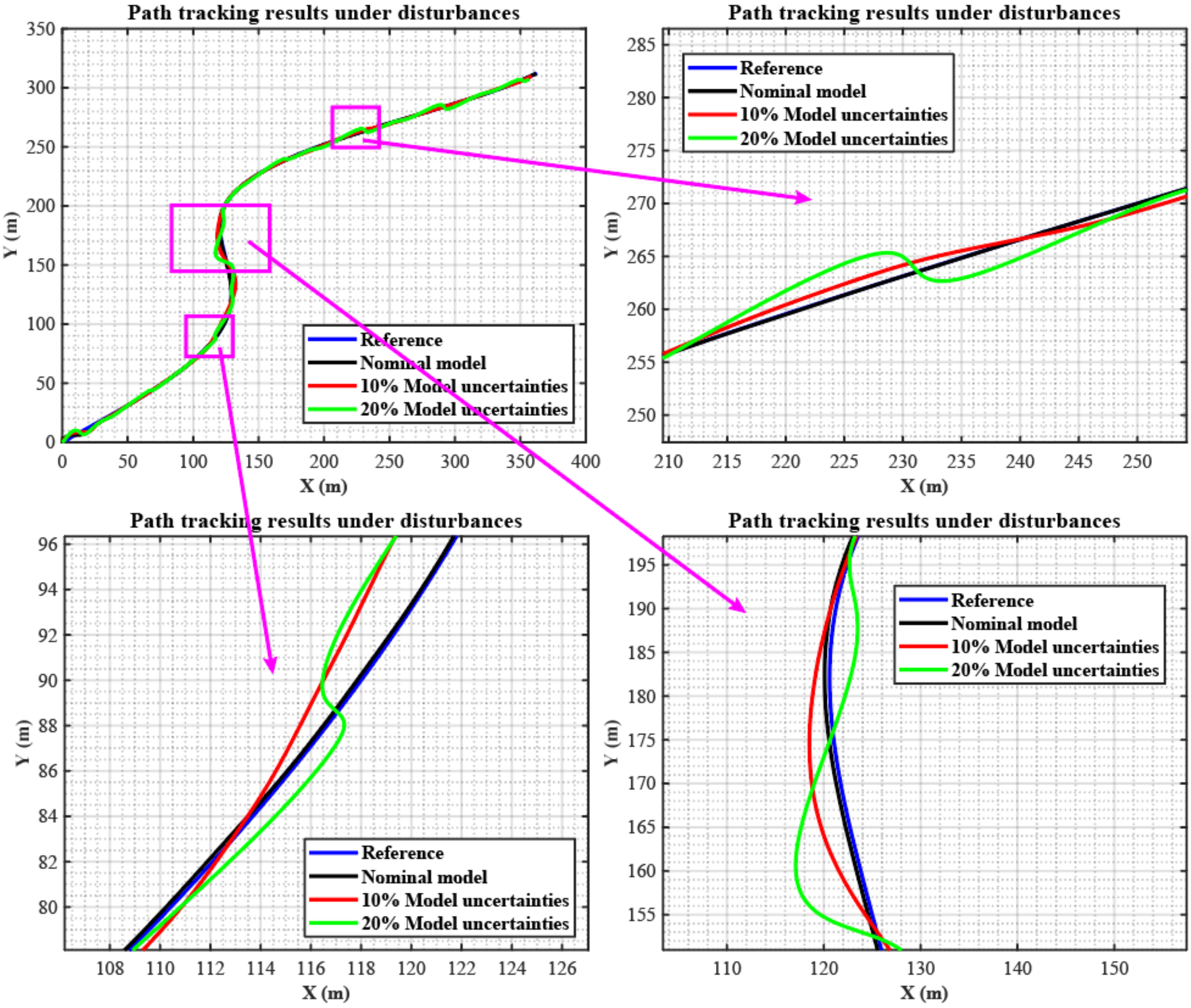

Fig. 15. Path tracking under different model uncertainties

The results are presented in Fig. 15, wherein the reference path is depicted in blue, the trajectory of the nominal model is illustrated in black, the model with 10% model uncertainty and disturbances is shown in red, and the green line indicates the model with 20% model uncertainty and disturbances. The effectiveness of the proposed algorithm in handling model uncertainty below 20% is evident from Fig. 15 and Fig. 16, where the USV successfully tracked the reference path with satisfactory results. The statistical analysis in Table 7 further confirms the performance of the algorithm, as evidenced by the integrated absolute error of (104.542, 102.218), (554.233, 369.489), and (803.098, 755.854) for nominal, 10%, and 20% models, respectively. Although some fluctuations are observed for relatively large model uncertainty, the USV demonstrates satisfactory performance in tracking the desired path, highlighting the robustness of the proposed algorithm.

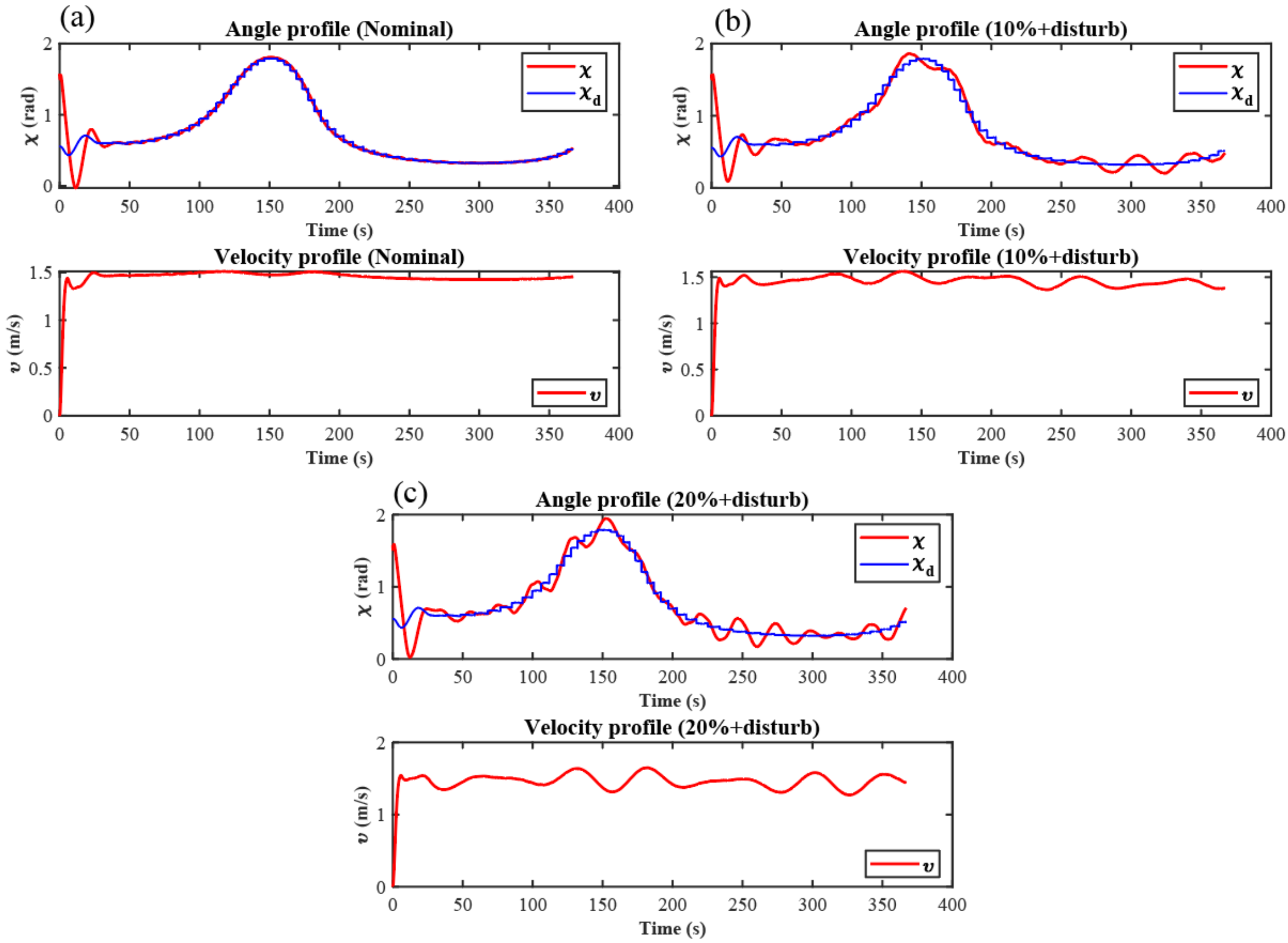

Fig. 16. Angle and velocity profile (a) Nominal model; (b) 10% model uncertainty and disturbances; (c) 20% model uncertainty and disturbances

Table 7. Evaluation indexes of control performance of NMPC

Case	IAE_x (m)	IAE_y (m)
Nominal	104.542	102.218
10% model uncertainty+disturb	554.233	369.489
20% model uncertainty+disturb	803.098	755.854

Note: for convenience, we recorded the error with the sampling frequency of 1 Hz.

4.2.2. Test 2: Comparative study with other methods

The Integral line of sight (ILOS) [41] and the adaptive LOS (ALOS) guidance [40] are implemented in combination with the standard proven-in-use PID controllers: the Otter vehicle employs a PID heading autopilot. The determination of the coefficients is set according to [50]: $K_P = 53.42$, $K_D = 14.84$, $K_I = 14.84$, $K_{FF} = 74.2$, where K_{FF} is the acceleration feed forward coefficient. The parameters and disturbances are set as the same in Section 4.2.1. Moreover, we choose the 10% model uncertainty in the comparative study. The reference path is designed as:

$$\begin{aligned} x_d &= 6.25\omega + 250\cos(1.5\pi\omega/40) - 0.05\omega^2 \\ y_d &= 8.75\omega - 0.05\omega^2, \end{aligned} \tag{30}$$

where ω is the path parameter that is independent of time. The USV is originally positioned at $x = 0\,m$, $y = 0\,m$, while the initial point of the path is assumed to be $\omega = 0$ and it ends at $\omega = 50$.

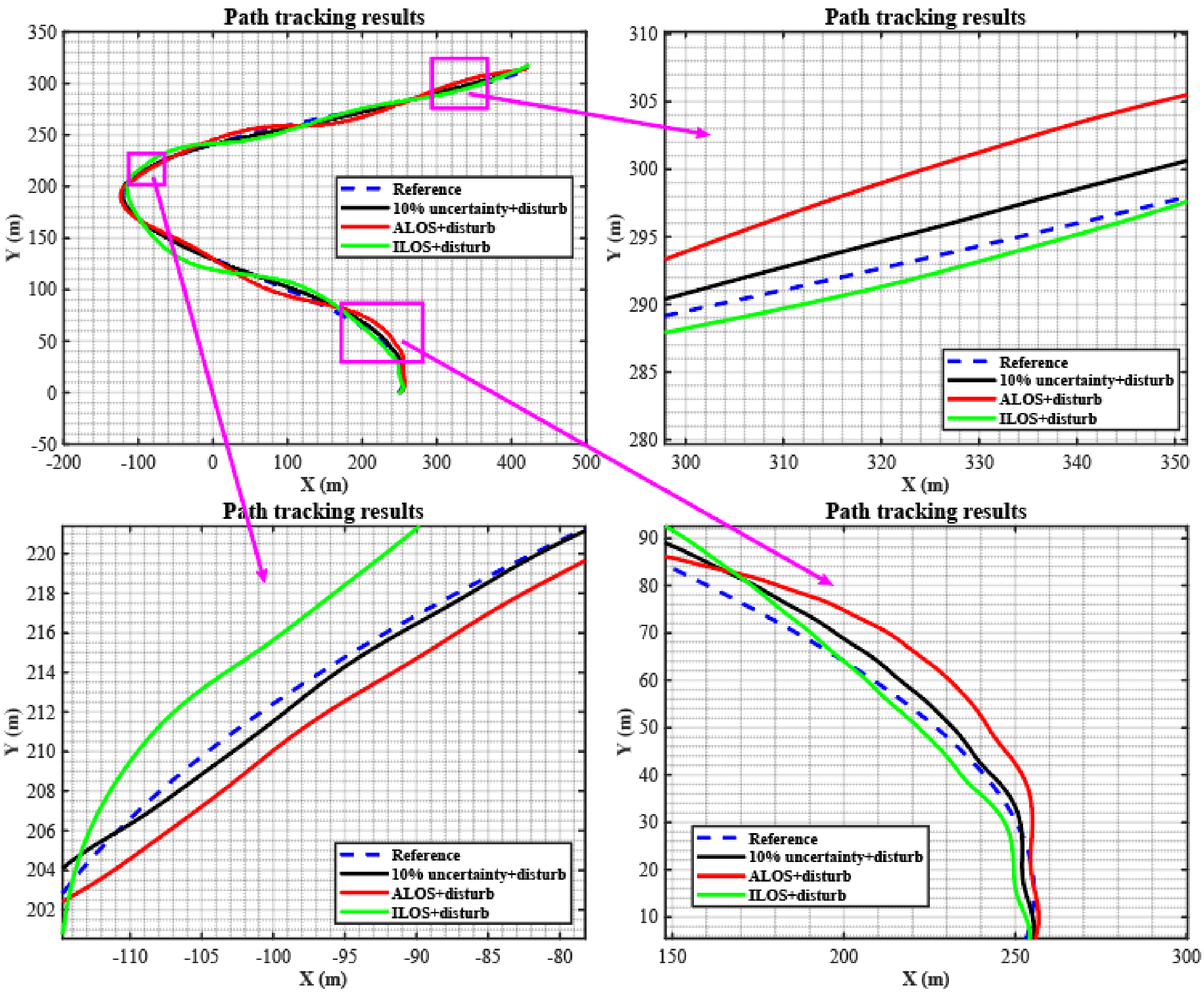

Fig. 17. Tracking results

Table 8 Evaluation indexes of control performance

Case	IAE_x (E+03 m)	IAE_y (E+03 m)
NMPC+10% uncertainty+disturb	1.7478	1.8453
ALOS+disturb	2.4022	2.2544
ILOS+disturb	2.7021	3.0882

Note: for convenience, we recorded the error with the sampling frequency of 1 Hz.

A comparative simulation between the proposed NMPC and the other references is shown in Fig. 17 and Table 8. The reference path is denoted by the blue dashed line, the trajectory of the NMPC is the solid black line, and ALOS and ILOS are represented by the red and green line, respectively. As shown in Fig. 17, It can be observed that the NMPC-based controller approaches the path and tracks the path more directly, incurring smaller tracking errors. However, it should be noted that the ALOS and ILOS methods exhibit a larger deviation from the reference path. This observation may be attributed to their comparatively weaker ability to reject external disturbances. Moreover, it cannot be overlooked that the suboptimal performance of the PID controller in the proposed framework, which may potentially result from inappropriate parameter tuning, could also have contributed to the observed discrepancies in tracking accuracy.

Observing the signal curves in Fig. 18 shows that the true course angle corresponds well with the reference signal in the case of NMPC. The other methods have shown relatively large deviations, especially for ILOS. As to the speed, time history curves are based on a constant design speed. It can be observed from the speed profile that NMPC yielded relatively stable velocity during the tracking, resulting in a smoother tracking performance. However, the speed profile demonstrates that the vehicle experiences a reduction in speed whenever it comes into contact with significant frequent deflections as a result of disturbances.

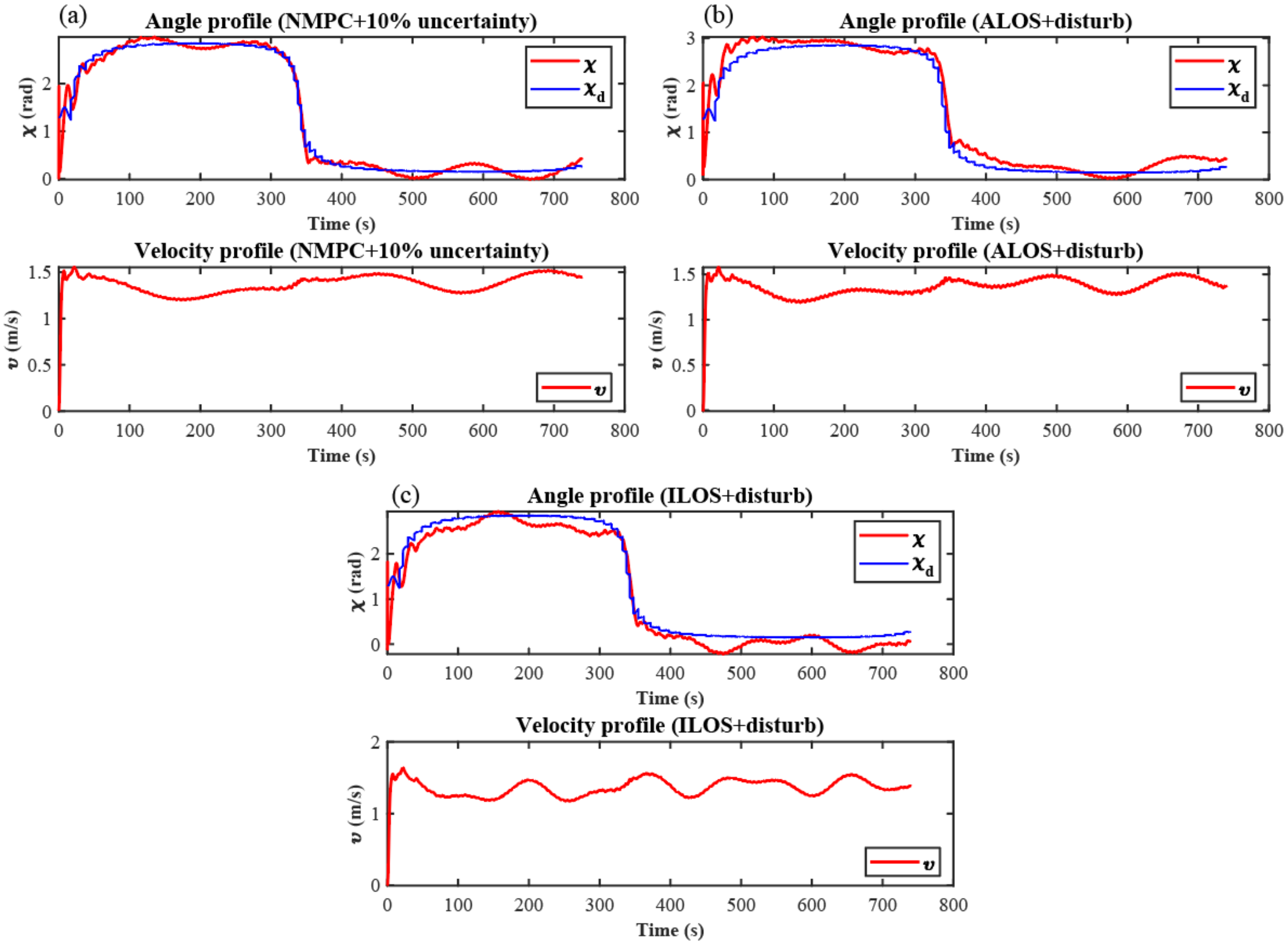

Fig. 18. Angle and velocity profile (a) NMPC; (b) ALOS; (c) ILOS

4.3. Simulation verification of the framework

In this section, the combination of the two modules is finally verified in a systematic way. The simulation is conducted under the context of a real-world water monitoring mission. We adopted the artificial lake at Zhejiang University's Zijingang Campus as the simulation site, see Fig. 19. We present the results with the local map, which has the origin of (120.076395°E, 30.299465°N) according to the satellite data. The local map has a maximum length and breadth of 501.6 m and 254.8 m, respectively.

As for the task distribution, 30 tasks are randomly distributed in the water environment, see Fig. 19. We assume the two USVs are the same but equipped with different sensors, e.g., USV1 is equipped with a conductive temperature depth (CTD) collector while USV2 carry the water collector, and these can be regarded exclusive tasks for the USVs. The common tasks are normal patrol missions that can be completed by both of the USVs. As denoted in Fig. 19, the common patrol tasks, CTD tasks for USV1, and water sampling tasks for USV2 are marked with black circles, red triangles, and green pentagrams, respectively. Each USV departs from its own base station and returns after completing the assigned tasks. The parameters of GPGA, NMPC, and the USV system are the same as in the previous sections.

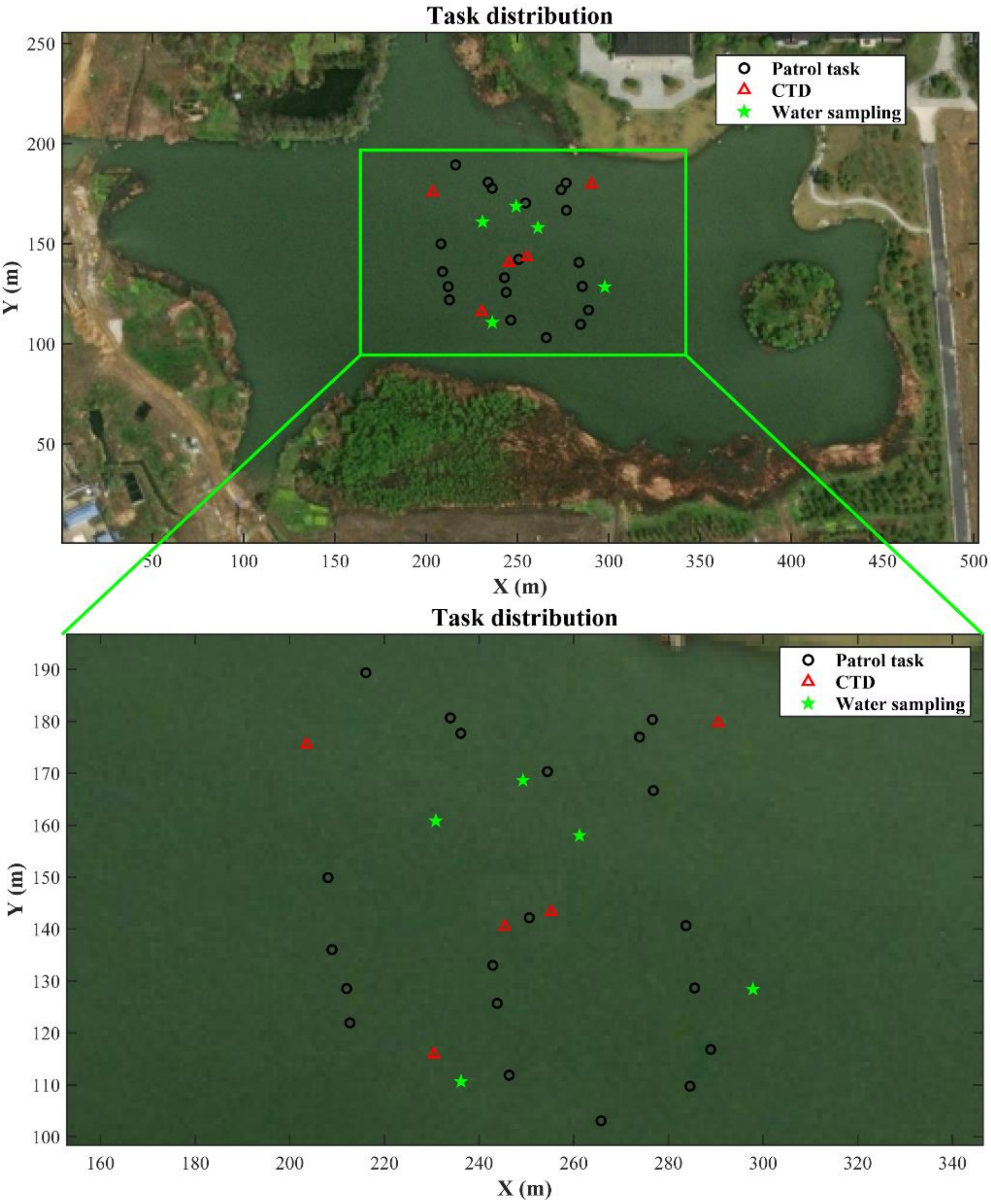

Fig. 19. Environment set

The ability of the proposed framework to consistently address the EMTSP and path following problem is eventually evaluated by conducting simulations under real-world geographies. In general, the proposed GPGA can optimally address the EMTSP with comparatively quick convergence performance. Fig. 20 presents the results of global path planning. As is shown in the figure, all the USVs with exclusive functional types have been successfully assigned their corresponding tasks. This is directly in line with our previous findings.

Fig. 21 shows the trajectories of the two USVs. It can be seen from the figure that the USVs have successfully completed the missions. With the aid of NMPC, the vehicles can autonomously reach all the planned points with satisfactory tracking performance. This indicates that the proposed framework can assist USV in performing water monitoring missions. Moreover, environmental loads and quick turns cause the vehicle's real trajectories to deviate slightly from the planned straight lines.

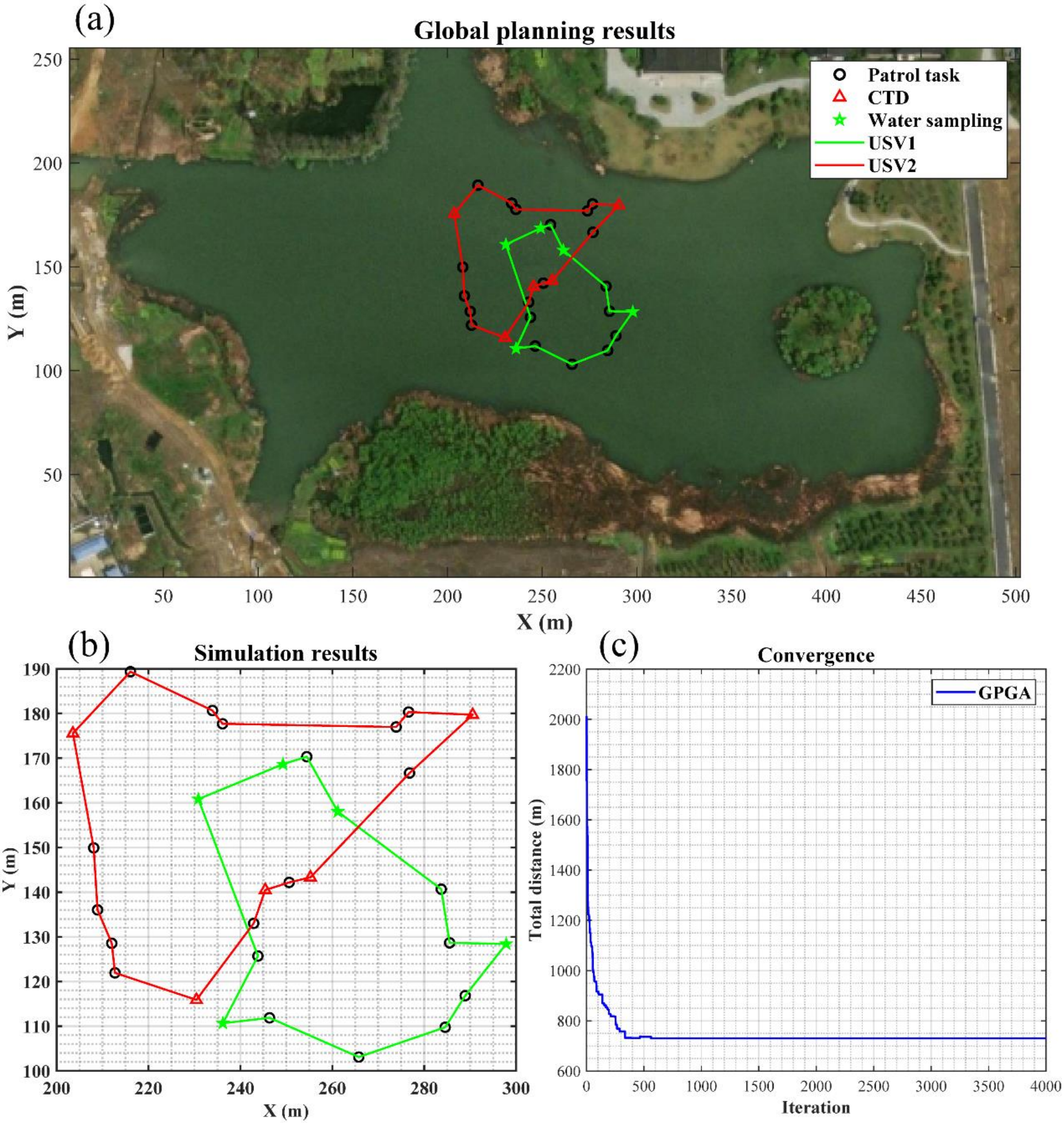

Fig. 20. Results of the global planning: (a) Path generation; (b) Path planned in the local frame; (c) Convergence history (time cost: 6.32 s)

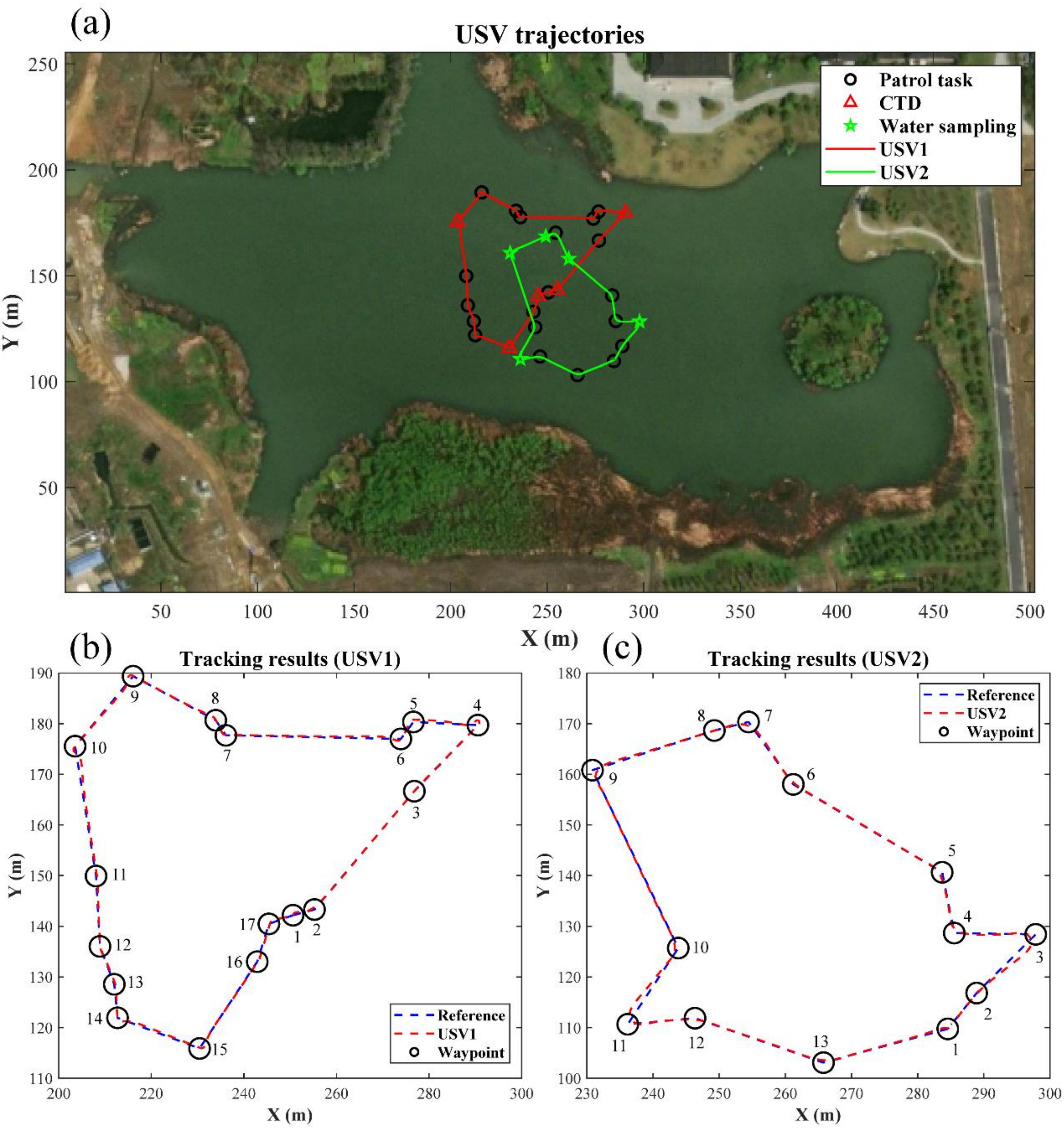

Fig. 21. Results of the waypoint following: (a) USV trajectories in satellite map; (b) Tracking results of USV1; (c) Tracking results of USV2

As to the tracking performance from the perspective of control, the USVs have shown rather satisfactory results in the path following mission. It is clear from comparing the two signal curves in Fig. 22 that the true heading angle matches well with the reference signal. However, a relatively large deviation exists when the vehicle passes through the point where the course changes significantly (around 30 s in Fig. 22. (a)). This is due to the sharp turnings. For the speed profile, time history curves are based on a constant design speed. Upon approaching the sharp turns, the surge speed was inconsistent. In spite of this, it remained close to the desired speed while following the course. Variations in the thruster signals reflected the vessel’s speed and direction as it turned.

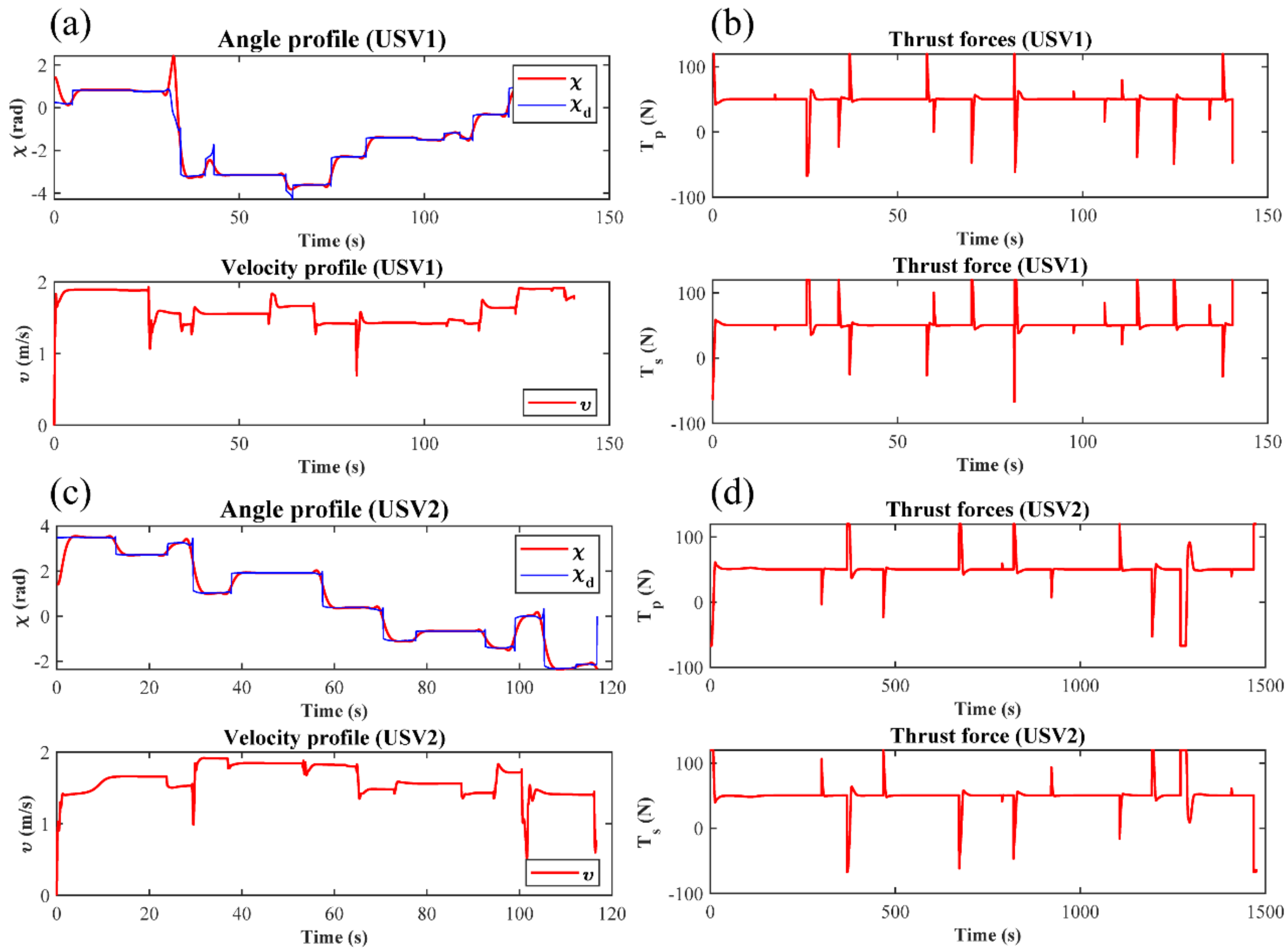

Fig. 22. Results of the tracking process: (a) Course angle and speed of USV1; (b) Thrust forces of USV1; (c) Couse angle and speed of USV2; (d) Thrust forces of USV2

5. Conclusion

The work in this paper presents an insightful study that focuses on path planning and path following for USVs in water monitoring missions. The particular class of global path planning consisted of problems where it is necessary to consider the heterogeneity of the USVs/tasks to complete the mission. Moreover, the inherent USV physical constraints pose a great challenge in achieving robust path following. This article creates a systematical approach against global path planning and path following with characteristics such as global optimality, rapid convergence rate, and robust control performance. From the corresponding results, it allows the following conclusions to be drawn:

- The presented results indicate the proposed EMTSP in combination with GPGA can consistently address the heterogeneous task planning of multiple USVs, thereby contributing to the water monitoring missions with specific needs.
- By utilizing the local exploration and greedy initialization, GPGA merits strong global searching ability and rapid convergence simultaneously. GPGA outperforms currently available combinatorial optimization approaches and provides improved solutions in all the problem variants.
- Finally, reference targets can be properly tracked by virtue of the NMPC strategy, ensuring smooth maneuvering by respecting USV physical constraints.

Some limitations of the current study need to be addressed in future work.

- Obstacles and unexpected invaders might threaten USV safety and potentially cause mission failure. This paper only deals with path planning in an obstacle-free area. In the future, efficient strategies can be applied to achieve obstacle avoidance.

- The presented NMPC solves a highly nonlinear optimization problem at each sample period, necessitating a significant processing and time capacity. Utilizing a more practical technique that can build a simpler version of the model permits the application of quadratic programming algorithms, resulting in a quicker implementation.
- The collision between the USVs is not considered. The authors are planning to design appropriate control strategies that could achieve coordination between the USVs.
- Moreover, the algorithm will be implemented in ROS systems and applied to actual USVs in a real-world water monitoring case.

Acknowledgement

The authors wish to sincerely thank the Editor-in-Chief, the associate Editor, and the anonymous referees for their comments and suggestions. This work is supported by the College of Civil Engineering and Architecture in Zhejiang University.

Competing interests

None.

References

[1] N. Wang, IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY 69 (2020).
[2] N. Wang, Y. Zhang, C.K. Ahn, Q. Xu, IEEE Trans. Veh. Technol. 71 (2022) 2358–2374.
[3] L. Zhao, Y. Bai, J.K. Paik, Ocean Engineering 280 (2023) 114750.
[4] L. Zhao, Y. Bai, F. Wang, J. Bai, Ships and Offshore Structures (2022) 1–13.
[5] L. Zhao, F. Wang, Y. Bai, Ships and Offshore Structures (2022) 1–10.
[6] S. Ma, W. Guo, R. Song, Y. Liu, Neurocomputing 420 (2021) 227–245.
[7] Y. Liu, R. Song, R. Bucknall, X. Zhang, Information Sciences 496 (2019) 180–197.
[8] Y. Liu, R. Bucknall, Neurocomputing 275 (2018) 1550–1566.
[9] H. Cao, Z. Guo, S. Wang, H. Cheng, C. Zhan, Water 12 (2020) 681.
[10] V. Nourani, N. Rouzegari, A. Molajou, A.H. Baghanam, Journal of Hydrology 587 (2020) 125018.
[11] A. Molajou, V. Nourani, A. Afshar, M. Khosravi, A. Brysiewicz, Water Resources Management 35 (2021) 2369–2384.
[12] Ö. Akbulut, A. Pakfiliz, Review of Computer Engineering Research 9 (2022) 222–238.
[13] W. Jin, Journal of Commercial Biotechnology 27 (2022).
[14] B. Xia, Journal of Commercial Biotechnology 25 (2022).
[15] Y. Li, X. Zhou, REVIEWS OF ADHESION AND ADHESIVES 10 (2022).
[16] W. Wang, Y. Wang, Journal of Commercial Biotechnology 27 (2022).
[17] H.-C. Chang, Y.-L. Hsu, S.-S. Hung, G.-R. Ou, J.-R. Wu, C. Hsu, Sensors 21 (2021) 1102.
[18] D. Madeo, A. Pozzebon, C. Mocenni, D. Bertoni, IEEE Trans. Instrum. Meas. 69 (2020) 1433–1444.
[19] S. Cryer, F. Carvalho, T. Wood, J.A. Strong, P. Brown, S. Loucaides, A. Young, R. Sanders, C. Evans, JMSE 8 (2020) 939.
[20] F. Zhang, O. Ennasr, E. Litchman, X. Tan, IEEE Systems Journal 10 (2016) 1271–1281.
[21] M.I. Sahalan, M.H.M. Idris, Z.Z. Abidin, M.A.A. Che Kamarudin, in: 2016 IEEE International Conference on Underwater System Technology: Theory and Applications (USYS), IEEE, Penang, 2016, pp. 48–54.
[22] K.L. Smith, A.D. Sherman, P.R. McGill, R.G. Henthorn, J. Ferreira, T.P. Connolly, C.L. Huffard, Science Robotics 6 (n.d.) eabl4925.
[23] J.H. Bae, B.-C. Min, S. Luo, S.S. Kannan, Y. Singh, B. Lee, R.M. Voyles, M. Postigo-Malaga, E.G. Zenteno, L.P. Aguilar, in: OCEANS 2019 MTS/IEEE SEATTLE, IEEE, Seattle, WA, USA, 2019, pp. 1–7.
[24] S. MahmoudZadeh, A. Abbasi, A. Yazdani, H. Wang, Y. Liu, Ocean Engineering 254 (2022) 111328.
[25] J. Zhong, B. Li, S. Li, F. Yang, P. Li, Y. Cui, Applied Ocean Research 111 (2021) 102658.

[26] Y. Gong, S. Zhang, M. Luo, S. Ma, Front. Neurorobot. 16 (2022) 1076455.

[27] A. Gonzalez-Garcia, I. Collado-Gonzalez, R. Cuan-Urquizo, C. Sotelo, D. Sotelo, H. Castañeda, Ocean Engineering 266 (2022) 112900.

[28] X. Zhou, X. Yu, Y. Zhang, Y. Luo, X. Peng, IEEE Trans. Automat. Sci. Eng. 18 (2021) 1575–1589.

[29] G. Tan, J. Zhuang, J. Zou, L. Wan, Applied Ocean Research 126 (2022) 103262.

[30] J. Xin, S. Li, J. Sheng, Y. Zhang, Y. Cui, Sensors 19 (2019) 3096.

[31] J. Xin, J. Zhong, F. Yang, Y. Cui, J. Sheng, Sensors 19 (2019) 2640.

[32] J. Xin, J. Zhong, S. Li, J. Sheng, Y. Cui, Sensors 19 (2019) 4620.

[33] X. Guo, M. Ji, Z. Zhao, D. Wen, W. Zhang, Ocean Engineering 216 (2020) 107693.

[34] S. Yang, J. Huang, W. Li, X. Xiang, JMSE 10 (2022) 1305.

[35] G. Xia, X. Sun, X. Xia, JMSE 9 (2021) 556.

[36] O. Cheikhrouhou, I. Khoufi, Computer Science Review 40 (2021) 100369.

[37] G. Bejarano, J.M. Manzano, J.R. Salvador, D. Limon, Ocean Engineering 258 (2022) 111764.

[38] A.M. Lekkas, T.I. Fossen, IEEE Trans. Contr. Syst. Technol. 22 (2014) 2287–2301.

[39] C. Paliotta, E. Lefeber, K.Y. Pettersen, J. Pinto, M. Costa, J.T. de Figueiredo Borges de Sousa, IEEE Trans. Contr. Syst. Technol. 27 (2019) 1423–1437.

[40] T.I. Fossen, K.Y. Pettersen, R. Galeazzi, IEEE Trans. Contr. Syst. Technol. 23 (2015) 820–827.

[41] W. Caharija, K.Y. Pettersen, M. Bibuli, P. Calado, E. Zereik, J. Braga, J.T. Gravdahl, A.J. Sorensen, M. Milovanovic, G. Bruzzone, IEEE Trans. Contr. Syst. Technol. 24 (2016) 1623–1642.

[42] B. Abdurahman, A. Savvaris, A. Tsourdos, Ocean Engineering 182 (2019) 412–426.

[43] B. Min, X. Zhang, Ocean Engineering 224 (2021) 108734.

[44] X. Yang, X. Yan, W. Liu, H. Ye, Z. Du, W. Zhong, Ocean Engineering 266 (2022) 112797.

[45] R. Cui, X. Zhang, D. Cui, Ocean Engineering 123 (2016) 45–54.

[46] N. Gu, D. Wang, Z. Peng, L. Liu, ISA Transactions 104 (2020) 212–221.

[47] W. Zhong, H. Li, Y. Meng, X. Yang, Y. Feng, H. Ye, W. Liu, Ocean Engineering 266 (2022) 112449.

[48] Y. Deng, X. Zhang, D. Zhao, T. Ni, M. Gong, Z. Zhang, Ocean Engineering 266 (2022) 113147.

[49] Z. Zheng, Y. Zou, ISA Transactions 65 (2016) 210–219.

[50] T. Fossen, Handbook of Marine Craft Hydrodynamics and Motion Control, 2011.

[51] M. Abdelaal, M. Fränzle, A. Hahn, Ocean Engineering 160 (2018) 168–180.

[52] L. Magni, G.D. Nicolao, L. Magnani, R. Scattolini, (2001).

[53] L. Grüne, J. Pannek, Nonlinear Model Predictive Control: Theory and Algorithms, 2011.

[54] Y. Xue, X. Wang, Y. Liu, G. Xue, in: 2021 7th International Conference on Mechatronics and Robotics Engineering (ICMRE), IEEE, Budapest, Hungary, 2021, pp. 150–155.

[55] H. Zhou, M. Song, W. Pedrycz, Applied Soft Computing 64 (2018) 564–580.

[56] J. Wang, O.K. Ersoy, M. He, F. Wang, Applied Soft Computing 43 (2016) 415–423.