Information-Driven Path Planning for Hybrid Aerial Underwater Vehicles

Zheng Zeng, Senior Member, IEEE, Chengke Xiong, Xinyi Yuan, Hexiong Zhou, Yuling Bai, Yufei Jin, Di Lu, and Lian Lian, Senior Member, IEEE

Abstract—This article presents a novel rapidly-exploring adaptive sampling tree algorithm for adaptive sampling missions using a hybrid aerial underwater vehicle (HAUV) in an air–sea 3-D environment. This algorithm innovatively combines the tournament-based point selection sampling strategy, the information heuristic search process, and the framework of the rapidly-exploring random tree algorithm. Hence, the vehicle can be guided to a region of interest to scientists for sampling and generate a collision-free path for maximizing information collection by the HAUV under the constraints of environmental effects of currents or wind and a limited budget. The simulation results show that the fast search adaptive sampling tree algorithm has higher optimization performance, faster solution speed, and better stability than the rapidly-exploring information gathering tree algorithm and the particle swarm optimization algorithm.

Index Terms—Autonomous robots, marine robots, path planning.

I. INTRODUCTION

Efficient observation of hydrometeorological parameters at the sea–air interface to obtain high-quality, high-resolution data could provide accurate boundary conditions for numerical prediction models of the ocean and atmosphere, which is essential for the study of physical mechanisms of sea–air interactions, accurate forecasting of typhoons (hurricanes), and marine disaster prevention and mitigation [1]. For the sea–air adaptive sampling mission, the observation platforms are desired to carry various sensors to acquire the ocean’s physical, chemical, and biological measurement data intelligently and autonomously in real time. Recently developed observation platforms including unmanned aerial vehicles (UAVs) [2], unmanned surface vehicles (USVs) [3], autonomous underwater vehicles (AUVs) [4], underwater gliders (UGs) [5], and other unmanned vehicles [6] are increasingly used by marine scientists for adaptive ocean observation and sampling. UAVs fly fast and can capture and measure changing atmospheric phenomena in the air. USVs sail fast, observe oceanic phenomena at the surface, and can carry small AUVs to a designated location for deployment if needed. AUVs are mobile when fully submerged, capable of observing and sampling small-scale oceanic phenomena underwater. UGs can work continuously underwater for several months to observe large and medium-scale ocean phenomena. However, the above-mentioned mobile observation platforms cannot simultaneously conduct joint sea–air observations of oceanic and atmospheric phenomena with 3-D distribution and high temporal and spatial variability in specific sea areas.

In recent years, a class of highly mobile hybrid aerial underwater vehicles (HAUV) [7], [8], which can conduct air, surface, and underwater surveys, has come into being. The HAUVs can carry air–sea optical observers and physicochemical sensors for air, surface, underwater detection, and data acquisition [9]. Compared with mobile observation platforms that can only operate in a single specific environmental medium, HAUVs have two modes of operation: aerial flight and underwater diving, so they can switch their modes of operation independently according to environmental information and mission requirements [10]; in addition, HAUVs have the advantages of higher mobility and lower operating costs [11] and can perform continuous, high-quality, and high-precision ocean and atmospheric characteristic parameters in the air–sea 3-D environment. This reduces the total cost of sea–air stereo observation and sampling and improves the operational efficiency and the amount of actual data acquisition in a single mission.

The use of HAUVs to perform sea–air stereo adaptive observation and sampling tasks can realize the simultaneous real-time observation of hydrometeorological data at the sea–air interface and provide a new observation means for multiscale sea–air interaction research as shown in Fig. 1. An HAUV could be launched from shore or a surface vehicle where upon a path-planning system could be used to generate a trajectory that leads the vehicle to a work site, perform a survey, and then return to shore or dock completely on its own. The surface vehicle will provide the HAUV with localization support, and acoustic localization is used to determine the HAUV’s position underwater. Typical miniature miniatur ultra-short baseline
To fulfill this mission, it is necessary for the path planner to consider the motion characteristics, the energy consumption of operation, and the time constraints of the HAUV in both ocean and atmosphere. Furthermore, due to the cross-field nature of sea–air interface observation, it is necessary to combine the definition and analysis methods of both ocean and atmospheric phenomena to determine the type, location, and time of the most exploitable hydrometeorological parameters in a sea area and plan feasible and optimal sampling paths [4]. In practical mission scenarios, joint sea–air stereo observation and adaptive sampling are directly associated with the path-planning system of the vehicle. The path-planning systems should determine the target observation area accessible by the HAUV based on the sea–air environment model and guide the HAUV to navigate along the optimal sampling path to collect the parameter data desired by marine scientists.

Therefore, the performance of the path-planning system is an important manifestation of the intelligence level of the HAUV for air–sea stereo observation, which determines the sampling efficiency and observation capability of the HAUV in the joint air–sea observation mission. The existing path-planning research mainly focuses on information-driven path planning for unmanned vehicles operating in a single medium, air, ground, or underwater [13], [14], [15]. There is an urgent need to develop a path-planning system applicable to the sea–air 3-D adaptive sampling of HAUVs. It should combine complex sea–air information with the motion performance and system constraints of HAUVs in a specific medium and adopt suitable path optimization algorithms so that the generated sampling paths can guide the HAUVs to the areas with rich environmental information for sampling.

Existing information-driven path-planning algorithms include branch-and-bound methods, random sampling algorithms, and population intelligence optimization algorithms. The random sampling algorithm and the classical rapidly-exploring random tree (RRT *) algorithm have low computational complexity and are suitable for solving path-planning problems in high-dimensional space. It can guarantee the probabilistic
completeness and asymptotic optimality of the solution. A review of existing information-driven path-planning algorithms is presented in Section II. This article will improve and expand on the RRT* algorithm by innovatively integrating the sampling strategy based on the tournament point selection method, the information heuristic search process, and the framework of the RRT* algorithm to design an algorithm applicable to information-driven path planning for HAUVs.

The main contributions of this work are listed as follows.

1. The information-driven HAUV path-planning problem is formulated under the constraints of the environmental effects of currents or wind and a limited budget.
2. A novel RAST path-planning method is presented for HAUV that can guide the vehicle to the region of interest for sampling and generate a collision-free path for maximizing information collection.
3. Several important optimization techniques applied to HAUV path planning have been compared in several scenarios, and the weaknesses and strengths of each optimization technique have been stated.

The rest of this article is organized as follows. In Section III, the information-driven path-planning problem for an HAUV is formulated. In Section IV, we propose the RAST* algorithm and design four different forms of the RAST* algorithm based on the theory of this algorithm and intend to investigate which optimization method can improve the computational speed and solution accuracy of the RAST* algorithm by introducing comparative experiments. In Section V, the classical rapidly-exploring information gathering tree (RIGT) algorithm and particle swarm optimization (PSO) algorithm are used as comparison algorithms, and the performance of different sampling path optimization algorithms is compared through simulation experiments. The prospect of applying RAST* algorithm in different scenarios is discussed. Concluding remarks are then presented in Section VI.

II. RELATED WORK

Information-driven path-planning is one of the key technologies for adaptive sampling of UAVs, ground vehicles, and AUVs [13], [14], [15], [16]. Take UAVs as an example, it aims to generate sampling paths that allow the UAV, within the constraints of a limited budget (e.g., energy and mission time), to maximize the amount of information observed and collected in the target area. Commonly used information-driven path-planning algorithms include branch-and-bound, population intelligence optimization algorithms, and random sampling algorithms.

A. Information-Driven Path Planning Based on Branch Delimitation Algorithm

The branch-and-bound method is widely used for solving constrained optimization problems and can efficiently search a finite number of feasible solution spaces systematically [17], [18], [19], [20]. Yilmaz et al. [21] proposed using the branch-and-bound method to solve the optimal sampling path for a single AUV and multiple AUVs to maximize the sum of line integrals of the uncertainty values along the entire path. Singh et al. [22] investigated the use of unmanned vehicles for sampling tasks with effective spatial coverage in application scenarios that require effective monitoring of spatiotemporal dynamic environments, such as water quality monitoring in rivers and lakes. Binney and Sukhatme [23] introduced a pilot measurement procedure in the branch-and-bound method to solve the adaptive ocean sampling path-planning problem of AUV. The authors verified that the branch-and-bound method with pilot measurement is more efficient than that without pilot measurement. Stankiewicz et al. [24] explored using AUVs with multiple sensors to build water quality models to help assess important watershed environmental hazards. The authors propose two information-driven path-planning algorithms, branch-and-bound and cross-entropy optimization, to select the future sampling locations of the AUV under the condition of the kinematic constraints of the AUV. The effectiveness of the proposed method is verified by simulation and field experiments. The branch-and-bound method is simple in structure and fast in solving and is suitable for information-driven path-planning problems in small-scale static environments, but leads to large memory space cost and computational burden.

B. Information-Driven Path Planning Based on Population Intelligence Optimization Algorithm

Population intelligence optimization algorithms, such as genetic algorithms (GA), PSO algorithms, and ant colony optimization (ACO) have been applied to information-driven path-planning problems in the literature. Heaney et al. [25], [26] used GA to plan adaptive sampling paths for multiple UGs. Arzamendia et al. [27] used GA to search for a water-free path that maximizes the sweeping area of the lake by a manned boat to monitor the lake environment. Frolov et al. [28] proposed a fully nonlinear GA for solving the optimal sampling path of AUV. The authors verified the optimization capability of GA by comparing it with the mowern method and A* algorithm through simulation experiments. Zhou et al. [29] integrated the fuzzy integrated evaluation method into a Multiobjective PSO algorithm to solve the path-planning problem of adaptive sampling with multiple AUVs in a dynamic ocean environment. This method uses sampling value and energy consumption as multi-objective cost functions. Xiong et al. [30] proposed an elite group-based PSO algorithm for planning AUV paths to maximize marine environmental feature information collection in a static ocean environment. Colmenares et al. [31] used an ACO algorithm to solve a single unmanned vehicle data collection task planning problem, planning a path for the vehicle to maximize the amount of water quality sampled. Xiong et al. [32] introduced the Delaunay space partitioning strategy into an ACO to form a path-planning system that can effectively guide the vehicle to the area of interest to the scientists. Hu et al. [33] used an ACO algorithm to plan the sampling path of AUV to maximize the acquisition of temperature data in the 3-D environment of temperature distribution of the regional ocean model system. However, only the simulation experiments of AUV in a 3-D unobstructed and current-free ocean environment
model were conducted. Other population intelligence optimization algorithms, including simulated annealing algorithm [34], [35], differential evolution algorithm [36], and covariance matrix adaptive evolution algorithm [37], [38], have also been applied to information-driven path-planning problems in the literature. The population intelligence optimization algorithm uses a population search model, simple theory, and easy application; however, as the dimensionality and size of the search space increases, the convergence speed decreases sharply, and it is easy to fall into local optimal solutions.

C. Information-Driven Path Planning Based on the Random Sampling Algorithm

One of the random sampling algorithms widely used in path planning is the rapidly-exploring random tree algorithm (RRT) [39]. Hollinger and Sukhatme [40] proposed a fast search information-gathering tree algorithm based on the RRT algorithm. It can plan paths that maximize information collection for AUVs with predefined constraints (e.g., energy or time constraints). It is also proved that the paths obtained after optimization are asymptotically optimal. Subsequently, Jadidi et al. [41] proposed an incremental search information collection tree algorithm based on this algorithm that can compute the sampling paths of USVs online. Cui et al. [42] proposed a multidimensional fast search random tree algorithm based on mutual information for solving multiple AUVs to maximize the understanding of the region of interest while minimizing the estimation error of the optimal sampling path. The authors verified the feasibility and effectiveness of the multidimensional search random tree algorithm based on mutual information through pool experiments. Viseras et al. [43] proposed a two-step path-planning strategy for robots to collect information about unknown physical processes efficiently. The RRT algorithm is used to determine the location points not yet visited by the robot in the first step and plan a sampling path that maximizes information collection while minimizing the path cost in the second step. Xiong et al. [44] incorporated a tournament selection method into the RRT algorithm and proposed an adaptive sampling-based path-planning system to generate an unmanned vehicle sampling path that maximizes information collection under the influence of obstacle environments and sea currents. The RRT algorithm has low computational complexity, which grows slowly as the size of the space increases. It is suitable for solving information-driven path-planning problems in a large high-dimensional space while guaranteeing the solution’s probabilistic completeness and asymptotic optimality. It can ensure the probabilistic completeness and asymptotic optimality of the solution.

Joint air–sea information-driven path planning is a high-dimensional, multiconstraint optimization problem. The computational effort and complexity of the problem will increase exponentially as the search space range increases, so the information-driven path-planning module for HAUVs needs to adopt an optimization algorithm that can quickly solve high-dimensional complex problems. Table I shows the current research status of the above three major classes of algorithms after synthesisization.

III. INFORMATION-DRIVEN PATH-PLANNING PROBLEM FOR HAUVS IN A 3-D ENVIRONMENT IN AIR AND SEA

The goal of the global information-driven path-planning system for HAUVs is to find a globally optimal sampling path \( \mathbf{P}^* \) from the set of feasible paths \( \mathbf{P}_\text{feas} \) that efficiently avoid obstacles \( C_{\text{obs}} \) (e.g., ships, reefs, and islands) and maximize the observation and collection of characteristic information (e.g., seawater temperature, salinity, chlorophyll fluorescence, dissolved oxygen concentration, air temperature, pressure, carbon dioxide fluxes, and turbulent heat fluxes) of interest to scientists. The impact of the wind and flow fields \( V_c \) on the vehicle should be fully considered.

Before discussing the problems studied in this article, the following assumptions are made.

Assumption 1: This article primarily focuses on a high-level planning architecture with simplified dynamics enabling it to find the optimum trajectory for maximizing information collection. Previous work has been done on studying the full dynamics of the system and the control strategies that drive the vehicle to the desired planned trajectories [45].

Assumption 2: The propulsion system of the HAUV maintains a constant thrust at economic power consumption, i.e., the vehicle maintains a constant flight speed \( V_{\text{air}} \) in the air, and a constant operation speed \( V_{\text{sea}} \) during the underwater navigation phase.

Assumption 3: The information map (IM) of environmental features studied in this article is given based on the actual observation needs of marine scientists, so the planning problem in this article is based on the prior known IM.

Remark 1: The information-driven path-planning problem studied in this article is formulated as follows: To study the

| Algorithm                      | Advantages                          | Disadvantages                                | Applicability                      |
|--------------------------------|-------------------------------------|----------------------------------------------|-----------------------------------|
| Branch-and-bound method        | Simple structure, fast solution speed| Linear, discrete search                      | Suitable for small scale static environment |
| Group intelligent optimization algorithm | Simple theory, easy to apply | Computation speed is greatly affected by space scale, easy to fall into the local optimum solution | Suitable for medium and small scale small and medium scale environment |
| Random sampling algorithm     | Low computational complexity, low speed of computation affected by spatial scale | Asymptotic optimization                     | Suitable for large scale High-dimensional environment |
characteristics of the sea–air interface in a specific sea area, an HAUV carrying limited energy $E_{\text{max}}$ is deployed from the deck of a research vessel. It is commanded to perform a sampling mission and at a specified mission time $T_{\text{max}}$ must return or land to another base station. According to Assumption 2, the velocity of the HAUV while flying in the air is $V_{\text{air}}$, the velocity while navigating underwater is $V_{\text{sea}}$, and the global sampling path is $\mathcal{P} = \{p_1, p_2, \ldots, p_n\}$, where $n$ is the number of discrete path points. In summary, the mathematical model can be established in the following form:

$$
\mathbb{P}^* = \arg \max f_\tau \left( (\text{IM}, V_{\text{air}}, V_{\text{sea}}, V_c, C_{\text{obs}}, E_{\text{max}}, T_{\text{max}}, \mathcal{W}) \right)
\text{ s.t. } V_{\text{air}} = 0, \quad V_{\text{sea}} = 0,
\forall i \in \{1, 2, \ldots, n\}, \quad \mathcal{P}_i \notin C_{\text{obs}}
E \leq E_{\text{max}}, \quad T \leq T_{\text{max}}
$$

where $f_\tau(\cdot)$ is the information collection function that returns the total amount of information collected along the entire path. $\mathcal{W}$ is the 3-D workspace of the HAUV. $E$ is the total energy consumption of the HAUV along the path $\mathbb{P}$ when performing the mission. $T$ is the total time of the HAUV along the path $\mathbb{P}$.

### A. Optimization Criterion

The optimization criterion of the HAUV information-driven path-planning problem is that the optimized path can maximize the information collected within a specific mission area and limited budgets. The information collection relies on the sensors onboard the HAUV, which can detect and collect data within a certain range of the current location $\mathcal{P}_i$, and build a 3-D array measured with all initial values of zero, the same size as the IM, which holds the values of collected environmental feature information. The HAUV performs the sampling task along the optimized path, constantly updating the information storage array measured[]. In addition, the value of feature information of different spatial locations in the air and sea environment may be variable, introducing the weight coefficient $\kappa$ into the information acquisition function. In summary, the total amount of information collected throughout the path can be expressed in the following form:

$$
f_\tau (\mathcal{P}) = \sum_{j=1}^{J} \kappa_j \cdot \text{measured} [\rho_j]
\text{ s.t. } \rho_j \in \mathcal{W}, \quad j \in \{1, 2, 3, \ldots, J\}
$$

where $\rho_j$ is the coordinate position of a raster point in the HAUV workspace. $\kappa_j$ is the information value weight of raster point $\rho_j$, and $J$ is the number of discrete raster points in the 3-D workspace.

### B. Sensor Models

HAUVs can carry a variety of sensors for aerial, surface, and underwater phenomena observation and information acquisition. Different sensors may have different sensing ranges and information acquisition capabilities, recent research makes use of continuous measurements of local ocean conditions from on-board current profiling sensors mounted in HAUV, e.g., an Oculus M750D forward looking sonar [46] and a StarFish sidescan sonar for information collection [47]. The Oculus M750D sensor is composed of 512 beams that allow aperture up to 120 m in front of the HAUV. In this article, the complexity of the sensor model is simplified. According to previous studies on various types of sensors, the ability of sensors to collect feature information decays with increasing distance [48], [49]. Therefore, the HAUV’s ability to collect information about the surrounding workspace at point $\rho_j$ can be expressed in the following form.

$$
\mathcal{A} (\mathcal{P}_1, \rho_j) = \begin{cases} 
\mathcal{A}_{\text{max}} e^{-\sigma \left(\frac{d_j}{\mathcal{A}_{\text{max}}}\right)^2}, & \text{if } d_j \leq d_{\text{max}} \\
0, & \text{if } d_j > d_{\text{max}}
\end{cases}
$$

where $d_j$ is the Euclidean distance between the two points $\mathcal{P}_1$ and $\rho_j$; $\mathcal{A}_{\text{max}}$, $\sigma$, and $d_{\text{max}}$ are the parameters of the sensor model, which control the sensing range and the sensor capability. Then, the sensor at the path point $\mathcal{P}_1$ can collect the amount of information at the point $\rho_j$ in the workspace as

$$
\text{sensor} (\mathcal{P}_1, \rho_j) = \text{IM}(\rho_j) \cdot \mathcal{A} (\mathcal{P}_1, \rho_j).
$$

The HAUV navigates along a planned path, and the information about environmental features is continuously collected by the sensors, which can continuously update the information storage array measured[] established in Section III-A. If the array measured[$\rho_j$] stores the information value of the collected raster points $\rho_j$ less than the information value collected by the sensor sensor($\mathcal{P}_1, \rho_j$), the array measured[$\rho_j$] is updated. Otherwise, it is not updated.

$$
\text{measured} [\rho_j] = \begin{cases} 
\text{sensor} (\mathcal{P}_1, \rho_j), & \text{if measured} [\rho_j] \leq \text{sensor} (\mathcal{P}_1, \rho_j), \\
\text{measured} [\rho_j], & \text{else}.
\end{cases}
$$

The update detection is carried out during the sampling task until the end of the path. Finally, the information collection function (1) returns the total amount of information that can be collected for the whole path. What needs to be noted in particular is that the focused object is not aimed at the specific amount of certain information, but the intensity of the change, such as the gradient of temperature and salinity. Obviously, the more drastic the feature changes, the more interesting phenomena are likely to occur. To further illustrate that the farther away from the rapidly changing area, the more limited information content that can be obtained, the sensor model in which the acquisition ability decays with the distance is introduced here.

### C. Constraint Conditions

Constraints of the HAUVs include the fact that the vehicles carry a limited amount of energy per mission and the possibility of prespecified mission times by marine scientists.

1) Energy Constraints: The HAUV has three different modes of motion during the sampling mission: airborne mode, underwater navigation mode, and cross-media mode. A complete
mathematical relationship between speed and energy consumption of the HAUV in these three modes of motion is not yet available. According to the existing literature, the relationship between speed and energy consumption of unmanned vehicles (e.g., UAVs and AUVs) can be analyzed. When the economic speed is maintained, the vehicle’s energy consumption per unit time can be minimized. Combining with Assumption 3, this article simplifies the mathematical model of the speed-energy consumption relationship of the HAUV, in which the power of the HAUV for economical air flight is known as \( P_{\text{air}} \), and the power for economic underwater navigation is \( P_{\text{sea}} \). The total power consumption of the HAUV can be expressed as the sum of the power consumption in the flight mode \( E_{\text{air}} \), the power consumption in the underwater navigation mode \( E_{\text{sea}} \), and the power consumption in the cross-media transition \( E_{\text{switch}} \).

\[
E = E_{\text{air}} + E_{\text{sea}} + E_{\text{switch}}
\]

(6)

\[
E_{\text{air}} = P_{\text{air}} \cdot T_{\text{air}}
\]

(7)

\[
E_{\text{sea}} = P_{\text{sea}} \cdot T_{\text{sea}}
\]

(8)

s.t. \( E \leq E_{\text{max}} \).

In addition, the total energy consumed by the HAUV per mission \( E \) cannot exceed the maximum energy \( E_{\text{max}} \). In (7) and (8), \( T_{\text{air}} \) and \( T_{\text{sea}} \) are the operating time of the HAUV in air flight mode and underwater navigation mode. The solution about the operating time will be explored in Section III-C2. It should be noted that the time and power consumption of a single cross-media transition is simplified to a constant.

2) Mission Time Constraints: Typically, marine scientists expect HAUV to complete sampling missions within a specified mission time. The total mission time can be expressed in the following form:

\[
T = T_{\text{air}} + T_{\text{sea}} + T_{\text{switch}} = \sum_{i=1}^{h-1} \frac{\| P_i - P_{i+1} \|}{V_{\text{abs},i}}
\]

(9)

s.t. \( T \leq T_{\text{max}} \) \( i \in \{1, 2, \ldots, h-1\} \)

where \( V_{\text{abs},i} \) is the actual operational velocity of the HAUV in the inertial coordinate system. The velocity of the HAUV is \( V_{\text{hauv}} \), including \( V_{\text{air}} \) in airborne flight mode, \( V_{\text{sea}} \) in the underwater navigation mode, and the variable speed motion in the cross-media transition mode. Considering the air–sea environment, there are wind and flow fields, and the actual operating velocity \( V_{\text{abs}} \) of the HAUV in the inertial coordinate system is affected by the environment. The actual operational velocity of the vehicle will be solved by the velocity vector synthesis method [50].

As shown in Fig. 2, the actual operational velocity direction of the HAUV on the path segment \( P_i, P_{i+1} \) should be consistent with the forward direction and the angle \( \cos \theta_i \) satisfies the following equation:

\[
\cos \theta_i = \frac{V_{c,i} \cdot V_{\text{abs},i}}{\|V_{c,i}\| \|V_{\text{abs},i}\|} = \frac{V_{c,i} \cdot a_i}{\|V_{c,i}\|}
\]

\[
= \frac{u_{c,i}a_{x,i} + v_{c,i}a_{y,i} + w_{c,i}a_{z,i}}{\sqrt{u_{c,i}^2 + v_{c,i}^2 + w_{c,i}^2}}
\]

(10)

where \( a_i \) is the unit vector of \( V_{\text{abs},i} \), that is, the unit vector of the path segment \( P_i, P_{i+1} \) whose components in the \( x, y, \) and \( z \) directions are \( a_{x,i}, a_{y,i}, \) and \( a_{z,i} \).

Suppose that the path-planning system has in advance the complete distribution information of the velocity field, including the velocity direction and magnitude. Then, the direction and magnitude of the velocity field \( V_{\text{abs},i} \), the direction of the actual operating velocity \( V_{\text{abs},i} \) of the HAUV in the inertial coordinate system, and the velocity generated by the thrusters in the airframe coordinate system \( V_{\text{hauv},i} \), according to the Law of Cosines

\[
V_{c,i}^2 + V_{\text{abs},i}^2 - 2V_{c,i}V_{\text{abs},i}\cos \theta_i = V_{\text{hauv},i}^2.
\]

(11)

The actual magnitude of the operational speed of the HAUV in the inertial coordinate system \( V_{\text{abs},i} \) can be deduced as the quadratic solution of the following equation:

\[
V_{\text{abs},i}^2 - 2(u_{c,i}a_{x,i} + v_{c,i}a_{y,i} + w_{c,i}a_{z,i})V_{\text{abs},i} + V_{c,i}^2 - V_{\text{hauv},i}^2 = 0.
\]

(12)

Let

\[
\Delta = 4(u_{c,i}a_{x,i} + v_{c,i}a_{y,i} + w_{c,i}a_{z,i})^2 + 4V_{\text{hauv},i}^2 - 4V_{c,i}^2
\]

when \( \Delta < 0 \), this equation has no real number solution, and when \( \Delta \geq 0 \), the solution of this equation can be expressed as

\[
V_{\text{abs},i} = u_{c,i}a_{x,i} + v_{c,i}a_{y,i} + w_{c,i}a_{z,i} \pm \frac{1}{2} \sqrt{\Delta}.
\]

(13)

When there are two feasible solutions, the positive solution with the larger value is usually chosen as the value of \( V_{\text{abs},i} \). In addition, when the value of the resulting solution is zero or negative, i.e., the value of \( V_{\text{abs},i} \) is not positive. This indicates that the path segment \( P_i, P_{i+1} \) is not reachable. A new feasible path needs to be generated. When \( \Delta < 0 \), it means the velocity component of the HAUV in the path segment \( P_i, P_{i+1} \) direction is not sufficient to counteract the velocity component of the flow or wind field in this direction, resulting in no real number solution to (12). Therefore, the algorithm must solve for the actual operating speed of the HAUV in each path segment, check whether the HAUV can reach each path point, and ensure that the optimized sampling paths are feasible.
D. Path Formation and Smoothing

The information-driven path-planning algorithm usually outputs a set of discrete path nodes \( \{p_1, p_2, p_3, \ldots \} \). To generate a path that satisfies the kinematics and dynamics of the HAUV, this article adopted B-spline curves for path smoothing [51, 52]. The principle of the B-spline curve is as follows. Assume the information-driven path-planning algorithm generates six path nodes with optimization points \( \{p_1, p_2, p_3, p_4, p_5, p_6\} \), where \( p_1 \) is the starting point and \( p_6 \) is the end point. These six path nodes are used as control points for the B-spline curve for curve fitting.

\[
\mathbf{P}(s_k) = \sum_{n=0}^{N} p_{k+n} \mathbf{B}_{n,N}(s_k) \quad \text{s.t. } s_k \in [0, 1], \ k \in [1, 2, \ldots, 6]
\]

where \( N \) is the order of the B-spline curve, and \( \mathbf{B}_{n,N}(s_k) \) is the Bernstein fundamental polynomial representing the B-spline basis function of the curve, which is defined as follows:

\[
\mathbf{B}_{n,N}(s_k) = \binom{N}{n} s_k^n (1-s_k)^{N-n} = \frac{n!}{n!(N-n)!} s_k^n (1-s_k)^{N-n}, \ n \in \{0, 1, \ldots, N\}
\]

where \( N = 3 \) and \( P(s_k) \) is continuously differentiable to second-order, i.e., the cubic B-spline curve generates a smooth path with continuous velocity and acceleration variation patterns from the start to the end of the HAUV. Therefore, the output optimized path \( \mathbf{P} = \{ \mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_h \} \) is continuous, smooth, and feasible.

E. Sea and Air 3-D Environment Modeling

The ocean and atmosphere environmental models can be obtained from official forecasts or built using analytical equations.

1) Forecast-Based 3-D Environment Model for Air and Sea: The distribution of 3-D environmental feature information usually conforms to a ternary Gaussian distribution. Assuming that there are \( B \) feature information regions of interest to marine scientists in the target area, the synthesized 3-D IM can be represented as a Gaussian mixture model

\[
\text{IM} = \sum_{b=1}^{B} g_b \cdot \mathcal{N}(\mu_b, \Sigma_b) \quad \text{s.t. } b \in \{1, 2, 3, \ldots, B\}
\]

where \( \mu_b = [x_b, y_b, z_b] \) is the mean value, which represents the position of the center of the feature information \( b \) in the workspace, \( \Sigma_b \) is a covariance matrix of size \( 3 \times 3 \), which controls the dispersion of the feature information \( b \) in the \( x, y, \) and \( z \) directions, and \( g_b \) is the weight parameter, which controls the peak size of the feature information \( b \).

In the simulation experiments of this article, the maps of the environmental feature information based on the analytic equations are created randomly by the Gaussian mixture model. The Gaussian distribution of feature information \( b \) in 3-D space is used as an example. A point in the workspace is randomly selected as the center \( \mu_b \) of feature information \( b \), and the randomly generated covariance matrix \( \Sigma_b \) is constructed in the following way.

1) First, construct a random diagonal matrix \( \mathbf{A} = \text{diag}(\mathbf{A}_x, \mathbf{A}_y, \mathbf{A}_z) \), where \( \mathbf{A}_x, \mathbf{A}_y, \mathbf{A}_z \) is random positive.
2) Create a random matrix \( \mathbf{B} \) and compute the standard orthogonal group of matrix \( \mathbf{B} = \text{orth}(\mathbf{B}) \).
3) If the characteristic value of the matrix \( C = \mathbf{B}^T \cdot c \mathbf{B} \) is greater than or equal to 0, the matrix is a randomly generated symmetric semipositive matrix, which can be used as a covariance matrix \( \Sigma_b \).

There are \( B \) randomly generated environmental characteristics in the workspace. Then, the information value of any point \( \rho_j \) can be defined as

\[
I(\rho_j) = \sum_{b=1}^{B} g_b e^{-\frac{1}{2}(\rho_j - \mu_b)^T \Sigma_b^{-1} (\rho_j - \mu_b)}. \quad (17)
\]

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
To have a unified measurement standard for different target sea areas, we used normalized processing between the \( I_{\text{air}} \) and \( I_{\text{sea}} \) in this article. It means the data distribution range of the feature information \( I_{\text{air}} \) and \( I_{\text{sea}} \) of all grid points in the workspace is [0,1].

According to the result discussed in Section III-E1, the 3-D velocity field can be decomposed into a set of different heights and depths of the 2-D horizontal speed field associated with each other. The speed field model at the 2-D level can be established and superimposed in a plurality of viscous Lamb eddy [54]. According to the analysis equation of the Lamb vortex, a Lamb vortex in the horizontal direction of the vertical position can be expressed as follows:

\[
V_{e_{x,y}} = f_c(R_i, \eta, \zeta)
\]

\[
u_e(R_i) = -\eta \frac{y - y_o}{2\pi (R_i - R_o)^2} \left[ 1 - e^{-\frac{(R_i - R_o)^2}{\zeta^2}} \right]
\]

\[
u_e(R_i) = -\eta \frac{x - x_o}{2\pi (R_i - R_o)^2} \left[ 1 - e^{-\frac{(R_i - R_o)^2}{\zeta^2}} \right].
\]

Among them,

\[
R_i = \begin{bmatrix} x \\ y \end{bmatrix}
\]

indicates a 2-D working space,

\[
R_i^0 = \begin{bmatrix} x_o \\ y_o \end{bmatrix}
\]

indicates the center position of the vortex, \( \eta \) represents vortex strength, and \( \zeta \) indicates the radius of the vortex. If there are multiple different locations, intensities, and radii of Lamb vortex in the field, the above three formulas are superimposed to solve the size and direction of the horizontal direction velocity field \( V_{e_{x,y}} \). The 3-D ambient speed field continuous gradient in the vertical direction is created by introducing the argument \( R_i^0, \eta, \zeta \).

Fig. 3 shows the maps of sea–air environment feature information generated based on the Gaussian mixture model and the velocity field generated by the Lamb vortex. The size of the raster in the schematic diagram is 100 × 100 × 13, indicating the searching area of 5 km × 5 km × 600 m, the gradient transparent red color indicates the information value of atmospheric features from 1 to 0, and the gradient transparent blue color indicates the information value of ocean features from 1 to 0. The translucent dark blue plane indicates the sea level. In this article, the wind field velocity is controlled within 5 m/s, and the flow field velocity is controlled within 0.4 m/s. From the top view in Fig. 3(b), we can see the velocity field in the vertical direction.

IV. INFORMATION-DRIVEN PATH-PLANNING ALGORITHM DESIGN

The objective of the RAST\(^{+}\) algorithm is to optimize the path of the HAUV, which maximizes the collection of air–sea environmental features and autonomously allocates the tasks of the HAUV in the air and underwater to meet the constraints of limited energy and preset mission time of the vehicle, and to avoid obstacles effectively. The RAST\(^{+}\) algorithm innovatively combines the sampling strategy based on the tournament point selection method, the information heuristic search process, and the RRT\(^{+}\) algorithm framework to achieve an efficient search of the air–sea IM to solve the optimal sampling path quickly. Four versions of the RAST\(^{+}\) algorithm are designed to verify the effectiveness and superiority of the RAST\(^{+}\) algorithm and study the effects of different optimization methods on the computational speed and solution accuracy of the RAST\(^{+}\) algorithm. The deformed rapidly-exploring random sampling tree\(^{+}\) (RRST\(^{+}\)) algorithm is designed according to the different sampling strategies. The RAST\(^{+}\)-I/E algorithm and the RAST\(^{+}\)-I algorithm are designed according to the information heuristic search process; the deformed rapidly-exploring adaptive sampling tree (RAST) algorithm is designed according to the presence or absence of information heuristic search and the reshaping process of parent nodes. Meanwhile, this article also uses the classical fast search information gathering tree algorithm and the PSO algorithm as comparison algorithms. The optimization process and technical details of these six algorithms are discussed in detail in the following.
A. Rapidly-Exploring Adaptive Sampling Tree Algorithm

The RAST* algorithm is a sampling-based algorithm inspired by the RRT* algorithm, but what differs from the RRT* algorithm is the introduction of a sampling strategy based on the tournament point selection method and an information heuristic search process in the main structure of the algorithm. Based on the RAST* algorithm, the sampling strategy tends to grow branches to the regions with high feature information values. The information heuristic search process searches for global sampling paths with low energy consumption during the iterative process of the algorithm, which helps to avoid the RAST* algorithm from falling into local optimum solutions. These two improvements enable the RAST* algorithm to generate the global optimal path for the HAUV. The pseudo-code of the RAST* algorithm is shown in Algorithm 1, and the main flow is as follows.

First, the parameters that need to be inputted before the RAST* algorithm can be executed include the following.

1) Environment model parameters $P_{\text{env}}$: information map IM, wind/currents field $V_c$, obstacle $C_{\text{obs}}$.
2) HAUV-related parameters $P_{\text{hauv}}$: the aerial speed $V_a$ and the operation power $P_{\text{aer}}$, underwater speed $V_w$ and the operation power $P_{\text{sea}}$, cross-media energy consumption $E_{\text{switch,1}}$ and time $T_{\text{switch,1}}$, limited energy $E_{\text{max}}$, sensor parameters $A_{\text{dmax}}, \sigma, d_{\text{max}}$.
3) Task-related parameters $P_{\text{task}}$: task start location $q_{\text{init}}$, end location $q_{\text{final}}$, preset mission time $T_{\text{max}}$.
4) RAST* algorithm parameters $P_{\text{cdrast}}$: the number of the predetermined raster points selected for the tournament $M$, step size $\delta$, neighboring radius $r$, maximum iterative number $\text{Max}_i$, the number of iterations the result is no longer improved $\text{It}_{\text{stop}}$.

Let $\text{Tree} = (\text{Vertex, Edge})$ denote the adaptive sampling tree. $\text{Vertex}$ is the set of tree nodes, and $\text{Edge}$ is the set of tree branch segments formed by node connections. $\text{Best}_I$G is a variable storing the optimal amount of information, and Bestsol is a 1-D array storing the optimal amount of information for each iteration.

The main structure of the RAST* algorithm has four key processes as follows, and the whole process is shown in Fig. 4.

Sampling strategy based on tournament point selection method (Algorithm 1, line 3): Compared with RRT*, the key improvement is to select the node with the highest information content from a randomly generated sequence of raster points from IM, rather than the traditional random selection. Deploying raster points with more information content as initial search nodes helps to grow branches to areas with high information values, which ultimately makes the path more focused on areas of interest. The TOURNAMENTSAMPLE() function in the steps shown in Algorithm 2 and Fig. 4(a), Raster points $\{q_1, q_2, \ldots, q_M\}$ are randomly selected from IM. Compare the information value in this group of $M$ raster points and return the location of the point $q_{\text{ts}}$ with the highest information content.

Search for the nearest tree node and generation of new nodes (Algorithm 1, lines 4 and 5): the nearest tree node is found in the set of tree nodes $\text{Vertex}$ around the sampling point $q_{\text{ts}}$. The search process of the nearest tree node is represented as the Nearest() function, and the specific steps are shown in Algorithm 3 and Fig. 4(b). Length(Vertex) function returns the number of tree nodes $v_n$ in $\text{Vertex}$. The Distance() function is used to solve for the Euclidean distance between two tree nodes, finding the index of the nearest tree node in the set $\text{Vertex}$, based on the 1-D array Dis. Use the tree node returned by the Nearest() function as $q_{\text{nearest}}$. Nearest() function can also be expressed as

$$\text{Nearest}(q_{\text{ts}}, \text{Vertex}) = \arg \min_{q_i \in \text{Vertex}} |q_{\text{ts}} - q_i|, \ i = 1, 2, \ldots, v_n.$$

(21)

Then, according to the steering function $\text{Steer}()$, grows from the tree node $q_{\text{nearest}}$ to the sampling point $q_{\text{ts}}$ with the tree branch whose length is $\delta$ and generate a new node $q_{\text{new}}$, as shown

---

**Algorithm 1: Rapidly-Exploring Adaptive Sampling Tree*.

**Enter:** 3D environment model parameters $P_{\text{env}}$, HAUV related parameters $P_{\text{hauv}}$, task-related parameters $P_{\text{task}}$, CDRAST algorithm parameters $P_{\text{cdrast}}$.

1: $\text{Vertex} = \{q_{\text{init}}\}; \text{Edge} = \emptyset; \text{Tree} =$ (Vertex, Edge); $\text{Best}_I$G = 0; Bestsol (1) = 0;

2: for $i = 1$ to $\text{Max}_i$ do

3: $q_{\text{ts}} \leftarrow \text{TournamentSample}(\text{IM}, M);$;

4: $q_{\text{nearest}} \leftarrow \text{Nearest}(q_{\text{ts}}, \text{Vertex});$

5: $q_{\text{new}} \leftarrow \text{Steer}(q_{\text{nearest}}, q_{\text{ts}}, r);$;

6: if $\text{CollisionFree}(C_{\text{obs}}, q_{\text{nearest}}, q_{\text{new}})$ then

7: $Q_m \leftarrow \text{Near}(\text{Vertex}, q_{\text{new}}, r);$;

8: $c_{\text{max}} = 0;$

9: for each $q_m \in Q_m$, do

10: $[\text{IG}, E, T, P] = \text{FitnessFun}(q_{\text{new}}, q_{\text{ts}}, q_{\text{min}}, \text{Vertex}, P_{\text{hauv}}, P_{\text{env}});$

11: $c_1 = \frac{\text{IG}}{P};$

12: if $c_1 \geq c_{\text{max}}$ & $E \leq E_{\text{max}}$ & $T \leq T_{\text{max}}$ & $\text{CollisionFree}(C_{\text{obs}}, P)$ then

13: $c_{\text{max}} = c_1; q_{\text{max}} \leftarrow q_m; P_{\text{max}} \leftarrow P;$

14: end if

15: end for

16: if CollisionFree($C_{\text{obs}}, P$) then

17: $q_{\text{new-parent}} \leftarrow q_{\text{max}},$

18: $q_{\text{new-T}} \leftarrow \text{Time}(P_{\text{max}});$

19: $q_{\text{new-E}} \leftarrow \text{Energy}(P_{\text{max}});$

20: $q_{\text{new-IG}} \leftarrow \text{IG}(P_{\text{max}});$

21: $\text{Vertex} \leftarrow \text{Vertex} \cup \{q_{\text{new}}\}$

22: $\text{Edge} \leftarrow \text{Edge} \cup \{(q_{\text{max}}, q_{\text{new}})\}$

23: if $q_{\text{new}}$ \text{IG} > Best\_\text{IG} then

24: Best\_\text{IG} = q_{\text{new-IG}}

25: end if

26: end if

27: end if

28: Bestsol(it) = Best\_\text{IG}

29: if Bestsol(it) - Bestsol(it) = Best\_\text{IG} then

30: break;

31: end if

32: end for

33: return Tree = (Vertex, Edge);

**Output:** $P^*, M_{\text{Best}_I}$G, Bestsol
Fig. 4. Illustration of RAST∗. (a) Sampling-based on tournament point selection method. (b) Search for nearest tree nodes. (c) Generation of new nodes. (d) Information heuristic search process. (e) Parent node reshaping.
Algorithm 2: Tournament Point Selection Function.
1: function TOURNAMENTSAMPLE(IM, M)
2: randomly selected M raster points \{q_1, q_2, \ldots , q_M\} from IM;
3: q_{ls} ← q_1;
4: for i = 2 to M do
5: if IM(q_i) > IM(q_{ls}) then
6: q_{ls} ← q_i;
7: end if
8: end for
9: return q_{ls};
10: end function

Algorithm 3: Nearest Point Selection Function.
1: function NEAREST (q_{ls}, Vertex)
2: vn = Length(Vertex);
3: for i = 1 to vn do
4: Dis(i) = Distance(q_i, q_{ls});
5: end for
6: [Dis_{min}, Index_{min}] = min(Dis);
7: return [q_{ls}, Index_{min}]
8: end function

Algorithm 4: Steer Function.
1: function STEER(q_{nearest}, q_{ls}, \delta)
2: D = Distance(q_{nearest}, q_{ls});
3: if D > \delta then
4: q_{new} = q_{nearest} + (q_{ls} - q_{nearest}) * \delta/D;
5: else
6: q_{new} ← q_{ls};
7: end if
8: return q_{new};
9: end function

Algorithm 5: Collision Detection Function.
1: function COLLISIONFREE(C_{obs}, varargin)
2: if varargin is not part of the C_{obs} then
3: return 1
4: else
5: return 0
6: end if
7: end function

Algorithm 6: Find the Set of Nearby Nodes Function.
1: function NEAR (Vertex, q_{new}, r)
2: Q_m ← \emptyset;
3: vn = Length(Vertex);
4: for i = 1 to vn do
5: if Distance(q_i, q_{new}) < r then
6: Q_m ← Q_m \cup q_i;
7: end if
8: end for
9: return Q_m;
10: end function

Algorithm 7: Adaptation Function.
1: function FITNESSFUN(q_{new},
2: q_{init}, q_{final}, Vertex, PA_{harn}, PA_{ex})
2: P = Connection(q_{new}, q_{init}, q_{final}, Vertex);
3: IG = I_r(P);
4: E = E_{Energy}(P);
5: T = E_{Time}(P);
6: return [IG, E, T, P]
7: end function

Information heuristic search process (Algorithm 1, lines 7–15): after checking those nodes, q_{nearest} and q_{new} are not in the obstacle space. Find the tree node whose distance between the node q_{new} and itself is less than r in the set of tree nodes Vertex according to the Near() function and stores it in the set of neighbors Q_m. The Near() function can also be expressed as

\[
\text{Nearest}(q_{ls}, \text{Vertex}, r) = q_i \in \text{Vertex} : |q_{ls} - q_i| \leq r, \quad i = 1, 2, \ldots , \text{vn}.
\] (23)

Use Connection() for each node in the neighborhood set Q_m. First retracts the parent node of q_m until it returns to the starting point q_{init} and then use a B-spline curve to fit the path segmentation from the starting point q_{init} to q_{new} and finally to q_{final}, until it forms the curvature continuous sampling path P. According to (1), (6), and (9), the total amount of collected information, the total energy consumption, and the total time consumption of path P can be solved. By cycling, the path that satisfies the constraints of energy, mission time, and no collision along with the most amount of information collected per unit power consumption is found as P_{max}, and the neighboring nodes that constitute this path are recorded q_{max}, as shown in Fig. 4(d).

Remodeling of the parent node and update of the optimal solution (Algorithm 1, lines 16–31): after checking that all discrete points in the path P_{max} are not within the obstacle space, q_{max} will be recorded as the parent of q_{new}, storing the total amount of collected information, total energy consumption and total time consumption of the path P constructed by the node q_{new}, and add nodes q_{new} to the set of tree nodes Vertex, segment (q_{max} and q_{new}) will be added to the tree branch set Edge as shown in Fig. 4(e). If the amount of collected information by the path generated in this iteration is greater than that of the previous iteration, then update

\[
\text{Steer}(q_{nearest}, q_{ls}, \delta) = \begin{cases} 
q_{nearest} + \frac{\delta}{q_{ls} - q_{nearest}} \cdot (q_{ls} - q_{nearest}), & \delta \leq |q_{ls} - q_{nearest}| \\
q_{ls}, & \delta > |q_{ls} - q_{nearest}|
\end{cases}
\] (22)
the variable Best_IG as the optimal solution for this iteration. Otherwise, do not update. After that, Best_IG is added to the array Bestsol. The algorithm keeps iterating in a loop until the optimal solution has no improvement after \( I_{\text{stop}} \) iterations. Otherwise, it iterates until the maximum number of iterations \( \text{Max}_{\text{it}} \) and outputs the path \( p^* \), the optimal solution \( \text{Best}_\text{IG} \) and the array of optimal solutions \( \text{Bestsol} \) generated by each iteration.

It is important to note here that the path-planning problem studied in this article has a large number of constraints. According to the characteristics of the HAUV model, if the mission start and end points are particularly far apart, there may be a situation where the HAUV is unreachable under the constraints of limited energy and mission time. In this case, the RAST* algorithm method may not be able to generate a feasible solution after \( I_{\text{stop}} \) iterations, and output Best_IG = 0. The optimized path \( p^* \) is empty. In this case, the mission starts and end positions need to be set again reasonably.

1) Different Improved Forms of the RAST* Algorithm: The RAST* algorithm uses \( c_{\text{max}} \) as the heuristic factor (line 11 of Algorithm 1). \( c_{\text{max}} \) is the amount of information collected by the path per unit power consumption to judge which node in the neighborhood set \( Q_m \) will be selected as the parent node of \( q_{\text{new}} \). Hence, this algorithm is named as RAST*-I/E algorithm in this article. The corresponding counterpart is the RAST*-I algorithm, whose heuristic factor \( c_{\text{max}} \) is the total amount of information collected throughout the path to judge which node in the neighborhood set \( Q_m \) will be selected as the parent node of \( q_{\text{new}} \). That is, the RAST*-I algorithm directly uses the target value as the value of \( c_{\text{max}} \). In theory, if the total amount of collected information is set as a heuristic factor and the power consumption as a constraint throughout the path, the initial search process may focus on acquiring more information in the short term without limiting power consumption. As a result, in the subsequent iterations, the tree nodes will not be able to grow to the region with high feature information value due to the lack of energy. Thus the RAST*-I algorithm tends to fall into local optimal solutions. The subsequent simulation experiments will compare the RAST*-I/E algorithm with the RAST*-I algorithm to analyze the information heuristic search process and analyze the influence of the selection of the heuristic factor \( c_{\text{max}} \) on the final results.

The framework of the RAST algorithm is consistent with that of the RRT algorithm. There is no heuristic search and parent node reshaping process in the optimization process and the tree node \( q_{\text{nearest}} \) is directly used as the parent node of \( q_{\text{new}} \), it means that lines 7–20 of Algorithm 1 are not executed. Although the RAST algorithm introduces a sampling strategy based on the tournament point selection method, leading the adaptive sampling tree to grow to regions with higher values, the RAST algorithm lacks inspiration. It does not reevaluate the fitness of the tree nodes with the newly generated nodes, so only feasible solutions can be obtained after optimization. The optimality of the RAST algorithm is the same as that of the RRT algorithm. Subsequent simulation experiments will compare the RAST*-I/E algorithm with the RAST algorithm to analyze the necessity and importance of applying the information heuristic search and the reshaping process of the parent nodes.

In addition, the RRST* algorithm is designed in this article depending on the sampling strategy. The sampling strategy of the RRST* algorithm is the same random sampling strategy as the RRT* algorithm, i.e., a point is randomly selected as the sampling point \( q_0 \) (line 3 of Algorithm 1), i.e., \( \mathbb{M} = 1 \); the rest of the RRST* algorithm procedure is the same as the RAST*-I/E algorithm. Theoretically, the advantage of the RRST* algorithm is that it maintains the randomness of the algorithm and can search and grow branches randomly. Hence, the RRST* algorithm explores the whole space more comprehensively. However, due to the stochastic nature of the RRST* algorithm, it requires more iterations to converge, leading to a longer computation time than the RAST*-I/E algorithm. Subsequent simulation experiments will also compare the RAST*-I/E algorithm and the RRST* algorithm to analyze the effect of different sampling strategies on the optimization results and efficiency.

2) Complexity Analysis of the RAST* Algorithm: Let \( n \) be the total number of iterations of the RAST* algorithm, and the main loop of the RAST* algorithm contains the iterations of the nearest tree node search process and the heuristic search process. The number of iterations of the Nearest() function in the nearest tree node search process is the number of iterations available so far. The number of iterations of the Near() function in the information heuristic search process is also the number of iterations available so far. Then, the time complexity of the processes is presented by the Nearest() function, so the time complexity of the RAST* algorithm and its deformation algorithm is \( O(n) \), the time complexity is \( O(n^2) \). However, according to existing studies in the literature [39], it is shown that the time complexity of the processes represented by the Nearest() function and the Near() function processes can be reduced to \( O(\log n) \) by certain methods, i.e., by reducing the number of loops. The nearest tree node position and the set of neighbors can be solved accurately while reducing the number of cycles. Therefore, the RAST* algorithm also can reduce the time complexity to \( O(n\log n) \).

The space complexity of the RAST* algorithm is defined as the amount of memory in the storage space for the adaptive sampling tree \( Tree = (\text{Vertex}, \text{Edge}) \), i.e., the size of the Tree set, i.e., \( \text{Size(Vertex)} + \text{Size(Edge)} \). In this article, the size of the Edge set does not exceed the total number of iterations of the RAST* algorithm, and the size of the Vertex set does not exceed the total number of iterations of the RAST* algorithm plus one because the initialization of the Vertex set already stores the starting point \( q_0 \). In summary, the RAST* algorithm has a maximum space complexity of \( O(n+n+1) \), i.e., \( O(n) \).

B. Rapidly Exploring Information Gathering Tree Algorithm

The RIGT algorithm is a sampling-based motion planning algorithm first proposed by Hollinger et al. for the information-driven path-planning problem [40]. The advantages of the RIGT algorithm have been analyzed in the literature as it can quickly search the entire workspace and reduce the number of branches and nodes in the tree collection by continuously growing and
Algorithm 8: Rapidly-exploring Information Gathering Tree

**Enter:** 3D environment model parameters $PA_e$, HAUV related parameters $PA_{haus}$, task-related parameters $PA_m$, RIGT* algorithm parameters $PA_{rigt}$.

1. Vertex $\leftarrow \{q_{init}\}$; Edge $\leftarrow \emptyset$; Tree $\leftarrow (\text{Vertex}, \text{Edge})$; $V_{closed} \leftarrow \emptyset$; Best $t_{IG} = 0$; Bestsol $(1) = 0$;
2. for $i = 1$ to Max_it do
   3. $c_{rand} \leftarrow \text{RandomSample}(I, M)$;
   4. $q_{nearest} \leftarrow \text{Nearest}(q_{rand}, \text{Vertex})$;
   5. $q_{feasible} \leftarrow \text{Steer}(q_{nearest}, q_{rand}, \delta)$;
   6. $Q_m \leftarrow \text{Near}(\text{Vertex}, q_{new}, 1)$;
   7. for each $q_m \in Q_m$ do
      8. $q_{new} \leftarrow \text{Steer}(q_m, q_{feasible}, \delta)$;
      9. if $\text{CollisionFree}(C_{obs}, q_{nearest}, q_{new})$ then
         10. $\{q_{new}, IG, q_{new}, E, q_{new}, T, P\} = \text{FitnessFun}(q_{new}, q_{init}, \text{limit}, q_{final}, \text{Vertex}, PA_{haus}, PA_e)$;
      11. if $\text{PRUNE}(q_{new})$ then
         12. Delete $q_{new}$
         13. else
            14. Vertex $\leftarrow$ Vertex $\cup \{q_{new}\}$
            15. Edge $\leftarrow$ Edge $\cup \{q_{max}, q_{new}\}$
            16. if $E > E_{\text{max}}$ or $T > T_{\text{max}}$ then
               17. $V_{closed} \leftarrow V_{closed} \cup \{q_{new}\}$
               18. else if $q_{new}.IG > \text{Best}_{IG}$ then
                  19. Best $IG = q_{new}.IG$
         20. end if
         21. end if
         22. end if
   23. Bestsol(it) $= \text{Best}_{IG}$
   24. if Bestsol(it) $-$ Bestsol(it $-$ It_stop) $= \text{Best}_{IG}$ break;
   25. end if
   26. end for
27. return Tree $\leftarrow (\text{Vertex}, \text{Edge})$;

**Output:** $P^*$, Best$_{IG}$, Bestsol

Pruning the tree collection, reducing the number of paths stored in the tree collection. The specific flow of the RIGT algorithm is shown in Algorithm 8.

Unlike the RAST* algorithm, the RIGT algorithm is a random sampling strategy, so the parameters of the RIGT algorithm $PA_{rigt}$ include only the step size $\delta$, the neighborhood radius $r$, the maximum number of iterations Max_it, and the optimization terminated if the solution no longer improved after It_stop iterations.

The basic idea of the RIGT algorithm is as follows.
1. Random sampling and nearest point search process: randomly generate sampling points $q_{rand}$ in the workspace and select the nearest sampling point $q_{nearest}$ nearest tree node $q_{rand}$. The branch grows with the length of $\delta$. Then, it forms a new node $q_{feasible}$ (Algorithm 8, lines 3–5).
2. Neighborhood access and tree set update process: find the set of neighbors $Q_m$, for each of the tree nodes in the set of neighbors $Q_m$, grows branches of length up to $\delta$ to the new node $q_{new}$. If neither this branch nor the new node is in the obstacle space and does not need to be cropped, they are put into the sets Edge and Vertex, respectively. Otherwise, delete the new node $q_{new}$; when the new node $q_{new}$ corresponds to a full path whose total energy consumption exceeds the maximum energy or total time consumption exceeds the specified mission time, the new node $q_{new}$ will be placed in the forbidden set $V_{closed}$. The tree nodes in this set no longer grow branches (Algorithm 8, lines 6–17).
3. Update the optimal solution process: update the optimal solution $\text{Best}_{IG}$ and the array of optimal solutions generated after each iteration Bestsol (Algorithm 8, lines 18–27).
4. Output results: after the main loop terminates, the optimal path $P^*$, the optimal solution Best$_{IG}$, and the array of optimal solutions Bestsol generated by each iteration is output.

The rules for whether the newly generated node $q_{new}$ by the RIGT algorithm in step 3 is cropped are defined in the literature as follows [36]. A node $q_{new}$ and its associated node $q_{m}$, if $q_{new}.IG < q_m.IG, q_{new}.E > q_m.E, q_{new}.T > q_m.T$, the nodes $q_{new}$ are trimmed.

### C. PSO Algorithm

The core idea of the PSO algorithm is as follows, $k$ particles are randomly generated as populations at initialization, and each particle represents a feasible solution. Let $p_k$ and $v_k$ be the position and velocity of the $k$th particle, respectively, and the PSO algorithm satisfies the following velocity and position update equations for the $k$th particle at the $i$th iteration:

$$v_{k}^{i+1} = w^i \cdot v_{k}^i + c1 \cdot \text{Rand1} \cdot (p_{pbest}^i - p_{k}^i) + c2 \cdot \text{Rand2} \cdot (p_{gbest}^i - p_{k}^i)$$

(24)

$$p_{k}^{i+1} = p_{k}^i + v_{k}^{i+1}$$

(25)

$w^i$ is the weight parameter at the $i$th iteration, and $c1$ and $c2$ are the learning factors. Rand$1$ and Rand$2$ are random numbers in the interval [0,1], $p_{pbest}^i$ is the $k$th particle optimal position at the $i$th iteration, and $p_{gbest}^i$ is the population optimal position at the $i$th iteration. It should be noted here that the weight parameter decays with an increasing number of iterations, satisfying the following equation:

$$w^{i+1} = w^i \cdot w_{\text{damp}}$$

(26)

where $w_{\text{damp}}$ is the decay rate of the weight parameter for each iteration. In the PSO algorithm, it is necessary to limit the maximum velocity of each particle in $x$, $y$, and $z$ directions $v_{\text{max}}$, to avoid too large a step; also, it is necessary to constrain that the updated position of each particle cannot overflow the workspace.

The PSO algorithm flow is shown in Algorithm 9, and the required input PSO algorithm parameters $PA_{pso}$ include the learning factors $c1$ and $c2$, initial weight coefficients $w^0$, the...
Algorithm 9: PSO Algorithm.

Enter: Environment model parameters $P_{A_e}$, HAUV related parameters $P_{A_{huv}}$, task-related parameters $P_{A_m}$, PSO algorithm parameters $P_{A_{ps}}$.

1: Initialize the position $p^0$ and velocity $v^0$ of each particle to ensure that each particle generates a feasible solution $IG^0_{gbest}$.
2: $IG_{gbest} = \max (IG^0_{gbest})$;
3: for $i = 1$ to $Max_{it}$ do
4: for $k = 1$ to $K$ do
5: Solving for $N_{ps}$ control points in particle $k$ according to (24) and (25), the $v_k^i$ and $p_k^i$;
6: The path nodes from the starting point through the control points to the endpoint are fitted with a B-spline curve to form path $P$;
7: $IG^i_k = f_{T_{time}}(P)$;
8: $E^i_k = f_{energy}(P)$;
9: $I^i_k = f_{time}(P)$;
10: if $IG^i_k > IG^{i-1}_{gbest,k}$ and $E^i_k > E_{max}$ and $I^i_k > T_{max}$ then;
11: $p_{gbest}^i = p_k^i$; $IG_{gbest} = IG^i_k$;
12: else
13: $p_{gbest}^i = p_{gbest}^{i-1}$; $IG_{gbest} = IG^{i-1}_{gbest}$;
14: end if
15: if $IG_{gbest} = IG^i_k$ then
16: $p_{gbest}^i = p_{gbest,k}^i$; $IG_{gbest} = IG^{i-1}_{gbest}$;
17: else
18: $p_{gbest}^i = p_{gbest}^{i-1}$;
19: end if
20: end for
21: $w^i = w^{i-1} \cdot w_{damp}$
22: Bestsol($i$) = $IG_{gbest}$
23: if Bestsol($i$) - Bestsol ($i$ - $I_{stop}$) = 0 then break;
24: end if
25: end for
26: end for

Output: $P_*$, $IG_{gbest}$, Bestsol

V. SIMULATION EXPERIMENT RESULTS AND ANALYSIS

In this section, the RAST$^*$-I/E algorithm, RAST$^*$-I algorithm, RAST$^*$ algorithm, RRST algorithm, and PSO algorithm designed in Section IV are compared in five simulation cases under various scenarios.

A. Simulation Experiment Setup

All simulation experiments in this section were performed on a host computer with Windows 10 operating system, Intel(R) Core(TM) i7-6700HQ CPU @ 3.40 GHz and 16.0 GB of RAM.

The parameter settings of the HAUV and all algorithms in the simulation experiments are shown in Table II. The parameters of the HAUV in this article are based on the values of the HAUV “Nezha” [55]. The optimal power of the HAUV for air flight $P_{air}$, the optimal power for underwater navigation $P_{sea}$ and single energy consumption for cross-media motion mode $E_{switch, 1}$, all three parameters are related to the HAUV with finite energy $E_{max}$. In this article, $E_{max}$ is used as a criterion to invert $P_{air}$, $P_{sea}$, and $E_{switch, 1}$ relative to the scale factor of $E_{max}$, thus determining the relative values of these three parameters. In addition, the basic parameter settings of all algorithms in this article are based on the summary of existing research literature. The step size $\sigma$ and the neighborhood radius $r$ of the RAST$^*$ algorithm are based on the size of the environmental raster map. In particular, it should be noted that the results of this article set $I_{stop}$ are set to 200 times to speed up the solution of the algorithm.

This section focuses on the simulation experiments of the information-driven path-planning problem for a single HAUV. The following five scenarios are designed.

Scenario 1: Path planning for an HAUV with limited energy.
Scenario 2: Path planning for an HAUV under the dual constraints of limited energy and mission time.
Scenario 3: Path planning for an HAUV under dual constraints of limited energy and tight mission time.
Scenario 4: Path planning for an HAUV with a higher weight of information on ocean features than on the atmosphere.
Scenario 5: Path planning for an HAUV with a higher weight of information on the atmosphere than the ocean.

The air and underwater environmental information weights are the same in Scenarios 1–3, and these simulations mainly demonstrate the performance of these six algorithms under different constraints. In Scenarios 4 and 5, different weights are set in the information collection function for the aerial and underwater features, and the simulations compare the optimization performance of these six algorithms.

The environmental model used in the above five scenarios is a $100 \times 100 \times 13$ raster map, representing a mission area of $5 \times 5 \times 600$ m. Each raster point in the raster map contains environmental feature information and velocity field data. The distribution of the values of all raster points in the workspace is $[0, 1]$. To evaluate the algorithm’s performance, several runs of experiments are performed for each algorithm, and the following performance indicators are introduced to measure and evaluate the strengths and weaknesses of the algorithm in terms of computational accuracy and efficiency.

1) The average information collection $I_{mean}$ satisfies the following equation:

$$I_{mean} = \frac{1}{N} \sum_{i=1}^{N} I_i$$

(27)

where $N$ is the number of samples, i.e., the times of experiment replications.

2) The standard deviation of the amount of information collected $I_{std}$, corresponding to the path generated in each
TABLE II  
PARAMETER SETTINGS FOR THE HAUV AND ALL ALGORITHMS

| Parameters                        | Notation | Value |
|-----------------------------------|----------|-------|
| HAUV                              |          |       |
| Air speed                         | $v_{\text{air}}$ | 10(m/s) |
| power for air flight              | $p_{\text{air}}$ | $\frac{1}{900} E_{\text{max}}$ |
| Underwater speed                  | $v_{\text{sea}}$ | 0.5(m/s) |
| Power for underwater navigation  | $p_{\text{sea}}$ | $\frac{1}{12880} E_{\text{max}}$ |
| Power consumption in cross-media  | $E_{\text{switch,1}}$ | $\frac{1}{30} E_{\text{max}}$ |
| Time consumption for cross-media  | $T_{\text{switch,1}}$ | 20(s) |
| Limited energy                    | $E_{\text{max}}$ | One standard unit |
| Sensor perception factor          | $\alpha_{d_{\text{max}}}$ | 1 |
| Sensor distance attenuation       | $\sigma$ | 1 |
| Sensing range                     | $d_{\text{max}}$ | 100(m) |
| RAST*Algorithm                    |          |       |
| Number of tournament selection    | $n$      | 10    |
| Step length                       | $\delta$ | 5     |
| Neighborhood radius               | $r$      | 10    |
| Maximum number of iterations      | $V_{\text{sea}}$ | 5000  |
| Number of times the result is no  | $t_{\text{stop}}$ | 200   |
| longer improved                   |          |       |
| PSO Algorithm                     |          |       |
| Learning factor                   | $c_{1}$  | 1     |
| Learning factor                   | $c_{2}$  | 1     |
| Initial weighting factor          | $c_{0}$  | 1     |
| Decay rate of weights             | $\omega_{\text{damp}}$ | 0.99  |
| Maximum particle velocity         | $v_{\text{pmax}}$ | [5.5,1] |
| Population size                   | $K$      | 50    |
| Number of control points          | $N_{\text{pnn}}$ | 5     |

repetitive experiment, quantitatively reflects the robustness of the algorithm. The standard deviation satisfies the following equation:

$$ I_{\text{std}} = \sqrt{\frac{1}{N-1} \sum_{i=1}^{N} |l_i - I_{\text{mean}}|^2}. \quad (28) $$

3) The average number of iterations.
4) The average computation time.

B. Information-Driven Path Planning for HAUV Under Different Constraints

In this section, we analyze the optimization performance of the above six algorithms to solve the HAUV information-driven path-planning problem under different constraints when the information weights of the sea and air environment features are the same.

1) Scenario 1: Path Planning for an HAUV With Limited Energy: First, consider an HAUV carrying a finite energy $E_{\text{max}}$, assume that the starting position of the HAUV is $q_{\text{init}} = (1 \text{ km, } 3.75 \text{ km, and } 0 \text{ m})$, the mission end position is $q_{\text{final}} = (4 \text{ km, } 3.75 \text{ km, and } 0 \text{ m})$, and the mission time is set to $T_{\text{max}} = \text{inf}$, the mission area is based on analytic equation, and there exists a submerged obstacle space similar to the continental slope. In this parameter setting, each algorithm is repeated ten times to verify the algorithm's stability, average optimization performance, and average computational speed.

Fig. 5 shows the HAUV sampling path, the distribution and convergence curve of environmental feature information collected along this path, and the error plot of information collection for ten repetitions of each algorithm. In the figure, the obstacle space is represented by a gray surface, the starting point is a yellow dot, and the endpoint is a green dot. The RAST*-I/E algorithm, RAST*-I algorithm, RAST algorithm, RRST*, RIGT, and PSO algorithms are orange, blue, red-brown, purple, lime green, and yellow-brown, respectively. From the graph of the optimized sampling paths of each algorithm, it can be seen that there is no limit on the RIGT algorithm performing aerial sampling in the pretask period. The main reason is that the sampling volume per unit energy consumption of the HAUV for underwater sampling is much higher than that for aerial sampling. In the absence of a mission time constraint, a global algorithm would choose to slowly collect the amount of information in the environment for a longer period to save the energy of the HAUV. However, the RIGT algorithm is relatively weak in global optimization, and the algorithm does not learn the experience of increasing the amount of information collected by extending the sampling time through iterations. Hence, the optimized path information collection is less.

The convergence curves in Fig. 5(g) show that the RAST*-I/E algorithm has the most information collection. The three algorithms with relatively slow convergence are the RAST*-I/E algorithm, the RRST* algorithm, and the RIGT algorithm, which require a relatively large number of iterations to find the asymptotically optimal solution. Fig. 5(h) shows the deviation of the results for each algorithm for ten repeated experiments. The two algorithms with larger deviations are RRST* algorithm and the RIGT algorithm, which can also be seen from the...
Fig. 5. Scenario 1: Informative path generated by RAST*-I/E (a), RAST*-I (b), RAST (c), RRST (d), RIGT (e), PSO (f) and convergence curve (g), error bar (h) produced by the path planners.
standard deviation in Table IV. The reason may be that these two algorithms’ sampling strategies are random, lacking good environmental information to guide the algorithm, which may repeatedly search near a local solution so that the results are no longer improved in a certain iterative process.

From the results recorded in Table III, we can find that, except for the RAST algorithm, the optimized paths of the other five algorithms almost exhaust the energy, meaning energy usage is maximized. The main reason is that, although the tournament-based point selection method can guide the adaptive sampling tree to grow toward regions with high values, the algorithm structure does not have the process of initiation search and parent node reshaping. The algorithm lacks initiation and does not reevaluate the adaptability of the tree nodes to the newly generated nodes. The simulation results show that the computation time of the RAST algorithm is significantly better than the rest of the algorithms. Still, it can only solve the feasible path, not the global optimal path, and thus the RAST algorithm has the lowest optimization performance.

From Tables III and IV, we can also obtain that the three algorithms with the best optimization performance are the RAST*-I/E algorithm, the RAST*-I algorithm, and the PSO algorithm. All three algorithms have an optimal information collection of more than 1000. However, it can also be seen in Fig. 5 that the paths generated by the RAST*-I/E algorithm are sampled back and forth underwater. In contrast, the PSO algorithm generates paths with the shape of S, and ultimately the PSO algorithm does not collect as much information as the RAST*-I/E algorithm.

Comparing the simulation results of the RAST*-I/E algorithm and the RAST*-I algorithm, the RAST*-I/E algorithm has slightly higher optimization performance than the RAST*-I algorithm but a slightly lower optimization speed. The only difference between these two algorithms is the heuristic factor in the information heuristic search process. The heuristic factor of the RAST*-I/E algorithm adopts the information collection amount directly, which means that the information heuristic search process of the RAST*-I algorithm does not consider the power consumption of the HAUV. On the other hand, the RAST*-I/E algorithm considers the power consumption of the HAUV, and the iterative search process is oriented toward collecting more information with less energy, but this increases the number of iterations. Therefore, the RAST*-I/E algorithm is not as fast as the RAST*-I algorithm, but its optimization performance is better.

2) Algorithm 2: Path Planning for an HAUV Under the Constraints of Limited Energy and Mission Time: In a real mission, marine scientists usually specify the mission time so that information can be obtained on time, which also facilitates the recovery of the HAUV. The environmental characteristics of the target area are downloaded from the official NOAA website for a small area in the Gulf of Mexico. Since the raster scale of the data provided by the official NOAA website is too large, this section scales the dataset to store the same feature information in a square environment of 50 m in length, width, and height for each raster. The dataset is normalized to $I_{air}$, $I_{sea} \in [0,1]$. Set the HAUV’s starting position as $q_{init} = (0.5 \text{ km}, 2.5 \text{ km}, 0 \text{ m})$ and the mission end position as $q_{fin} = (4.5 \text{ km}, 2.5 \text{ km}, 0 \text{ m})$, the mission time is $T_{max} = 3 \text{ h}$.

From Fig. 6, it can be seen that the RAST*-I/E algorithm has the least remaining red and blue areas compared to other algorithms. It means that the RAST*-I/E algorithm captures the most information, which is quantitatively shown in Fig. 6(g) and Table V. Combining Fig. 6(h) and Table VI, it can be found that the three algorithms with the best optimization performance are still the RAST*-I/E algorithm, RAST*-I algorithm, and PSO algorithm.

The optimal results in Table V show that the path generated by the RAST*-I/E algorithm and the RAST*-I algorithm used almost all the power and mission time. The RRST, RIGT, and PSO algorithms reach the boundary of one single constraint and have a fair search capability, whereas the RAST algorithm still only generates feasible solutions. The performance metrics in

| Algorithm | Average message size | Standard deviation | Average number of iterations | Average computation time (s) |
|-----------|----------------------|--------------------|-----------------------------|-----------------------------|
| RAST*-I/E | 1077.20              | 84.01              | 1401                        | 52                          |
| RAST*-I   | 963.08               | 58.07              | 951                         | 30                          |
| RAST      | 485.21               | 28.79              | 611                         | 3                           |
| RRST*     | 734.70               | 171.23             | 1294                        | 43                          |
| RIGT      | 656.06               | 121.73             | 1360                        | 34                          |
| PSO       | 912.42               | 108.99             | 828                         | 46                          |
Fig. 6. Scenario 2: Informative path generated by RAST*-I/E (a), RAST*-I (b), RAST (c), RRST (d), RIGT (e), PSO (f) and convergence curve (g), error bar (h) produced by the path planners.
Table VI still reflect a similar situation to Scenario 1, i.e., the random sampling strategy makes the RRST∗ algorithm and the RIGT algorithm require more iterations to find the global solution. The RAST∗-I/E algorithm performs better under multiple constraints than one single constraint.

3) Scenario 3: Path Planning for an HAUV Under the Constraints of Limited Energy and Tight Mission Time: While the mission time studied in Scenario 2 is more relaxed, this scenario will conduct experiments for path planning of an HAUV under a tighter mission time. It is assumed that the HAUV needs to be recovered at this location after 1 h. As shown in Fig. 7, the areas with high information values of the atmosphere are distributed in the range of [3 km, 5 km] on the x-axis. The areas with high information values of oceanic features are distributed in the range of [1 km, 3 km] on the x-axis. Due to the tight mission time, the HAUV prefers to collect atmospheric feature information, but it is constrained by the limited energy to perform only aerial sampling. The optimized path maximizes atmospheric and oceanic information collection under double constraints. The convergence curves in Fig. 7(g) show that the RRST∗ algorithm based on the random sampling strategy and the RIGT algorithm has the highest convergence. The RIGT algorithm converges the slowest. The information acquisition error in Fig. 7(h) shows that the optimization capability of the algorithms in this example can be roughly divided into three echelons. The first echelon is the RAST∗-I/E algorithm, the second echelon is the RAST∗-I algorithm, RRST∗ algorithm, and PSO algorithm, and the third echelon is the RAST algorithm and RIGT algorithm.

From Table VII, we can calculate that the optimal result of the RAST∗-I/E algorithm has 12% higher information collection than the RRST∗ algorithm. The reason is that the RAST∗-I/E algorithm introduces the tournament point selection method into the sampling strategy. Therefore, more sampling points fall in the regions where feature information is gathered. From Table VIII, it can be calculated that the average information collection of the optimized results of the RAST∗-I/E algorithm is 19% higher than that of the second. The simulation results of this example fully verify the optimization performance and stability of the RAST∗-I/E. However, the speed of the RAST∗-I/E algorithm is only moderate. In addition, the average computation time of the PSO algorithm in Table VIII is higher than that of the remaining five algorithms, mainly because the process of finding feasible solutions for the initial particle swarm is uncertain.

C. Information-Driven Path Planning for an HAUV With Weighted Environmental Feature Information

This section analytically investigates the performance of the above six algorithms for solving the HAUV in different weights of information on the sea and air environment.

1) Algorithm 4: Path Planning for an HAUV With Higher Weight of Ocean Feature Information Than the Atmosphere: It is
Fig. 7. Scenario 3: Informative path generated by RAST*-I/E (a), RAST*-I (b), RAST (c), RRST (d), RIGT (e), PSO (f) and convergence curve (g), error bar (h) produced by the path planners.
TABLE VIII
SCENARIO 3: REPEATED EXPERIMENTAL ALGORITHM PERFORMANCE METRICS COMPARISON

| Algorithm   | Average message size | Standard deviation | Average number of iterations | Average computation time (s) |
|-------------|----------------------|--------------------|-------------------------------|-----------------------------|
| RAST*+I/E  | 620.33               | 48.60              | 999                           | 45                          |
| RAST*+I    | 511.98               | 41.10              | 831                           | 32                          |
| RAST        | 231.18               | 18.43              | 564                           | 2                           |
| RRST*       | 513.00               | 63.02              | 1351                          | 41                          |
| RIGT        | 234.90               | 87.40              | 695                           | 26                          |
| PSO         | 521.01               | 60.59              | 750                           | 63                          |

TABLE IX
SCENARIO 4: COMPARISON OF THE OPTIMAL RESULTS OF THE SIMULATION EXPERIMENT ALGORITHM

| Algorithm   | Amount of information collected | Iteration times | Energy consumption ($E_{max}$) | Task execution time (h) |
|-------------|---------------------------------|-----------------|--------------------------------|-------------------------|
| RAST*+I/E  | 2914.08                         | 745             | 1.00                           | 7.47                    |
| RAST*+I    | 2234.58                         | 1148            | 1.00                           | 6.21                    |
| RAST        | 1288.18                         | 458             | 0.94                           | 4.09                    |
| RRST*       | 1875.60                         | 959             | 0.90                           | 6.67                    |
| RIGT        | 2331.55                         | 2493            | 1.00                           | 6.89                    |
| PSO         | 2641.41                         | 915             | 1.00                           | 7.48                    |

TABLE X
SCENARIO 4: REPEATED EXPERIMENTAL ALGORITHM PERFORMANCE METRICS COMPARISON

| Algorithm   | Average message size | Standard deviation | Average number of iterations | Average computation time (s) |
|-------------|----------------------|--------------------|-------------------------------|-----------------------------|
| RAST*+I/E  | 2841.31              | 87.36              | 1008                          | 54                          |
| RAST*+I    | 2147.38              | 74.94              | 866                           | 46                          |
| RAST        | 1132.68              | 86.04              | 528                           | 2                           |
| RRST*       | 1648.89              | 139.46             | 987                           | 45                          |
| RIGT        | 1686.39              | 416.07             | 1532                          | 69                          |
| PSO         | 2569.83              | 40.51              | 947                           | 94                          |

assumed that marine scientists are more interested in information about the ocean, assigning a weighting factor of ocean feature information weighting factor $\kappa_{sea} = 3$ and the weighting factor of atmospheric feature information $\kappa_{air} = 1$.

As shown in Fig. 8, each algorithm crosses the obstacle space and travels to the region where environmental feature information is gathered. Due to the high weight of ocean information, algorithms with high optimization capabilities focus on collecting underwater information. This conclusion is corroborated in Fig. 8, where only the RAST algorithm activates the cross-media motion mode for the HAUV. However, the optimized path does not capture more environmental information, so the RAST algorithm is still not able to search for the global solution. From Fig. 8(g) and (h), it can be seen that the optimization speed and stability of the RIGT algorithm are poor. According to the convergence curves Fig. 9, it is found that the PSO algorithm and RAST*+I/E algorithm generate the path with maximum information collection. In Tables XI and XII, the optimal results and the performance indexes show that both algorithms perform well in repeated experiments. The optimization performance of the PSO algorithm is slightly higher than that of the RAST*+I/E algorithm, and the number of iterations is reduced by nearly two times, which highlights that the PSO algorithm is indeed an excellent global optimization algorithm. However, the standard deviation of the PSO algorithm in repeated runs is higher than that of the RAST*+I/E algorithm, indicating that the PSO algorithm is not as stable as the RAST*+I/E algorithm. Moreover, the average computation time of the PSO algorithm is slightly longer than that of the RAST*+I/E algorithm, indicating that a large amount of time is wasted in finding the initial feasible solution. Still, once the initial feasible solution of the particle swarm is generated, the convergence speed of the algorithm’s main loop can be accelerated, thus

2) Scenario 5: Path Planning for an HAUV With a Higher Weighting of Atmospheric Feature Information Than the Ocean: Assuming that marine scientists are more interested in atmospheric information, a weighting factor $\kappa_{air} = 3$ is assigned to atmospheric information and a weighting factor $\kappa_{sea} = 1$ to oceanic information.

From the paths of each algorithm shown in Fig. 9, it can be found that, except for the RAST algorithm, all the other five algorithms only perform aerial sampling before heading to the endpoint, which is due to the high weight of atmospheric feature information.

According to the convergence curves Fig. 9(g), it is found that the PSO algorithm and RAST*+I/E algorithm generate the path with maximum information collection. In Tables XI and XII, the optimal results and the performance indexes show that both algorithms perform well in repeated experiments. The optimization performance of the PSO algorithm is slightly higher than that of the RAST*+I/E algorithm, and the number of iterations is reduced by nearly two times, which highlights that the PSO algorithm is indeed an excellent global optimization algorithm. However, the standard deviation of the PSO algorithm in repeated runs is higher than that of the RAST*+I/E algorithm, indicating that the PSO algorithm is not as stable as the RAST*+I/E algorithm. Moreover, the average computation time of the PSO algorithm is slightly longer than that of the RAST*+I/E algorithm, indicating that a large amount of time is wasted in finding the initial feasible solution. Still, once the initial feasible solution of the particle swarm is generated, the convergence speed of the algorithm’s main loop can be accelerated, thus
Fig. 8. Scenario 4: Informative path generated by RAST∗-I/E (a), RAST∗-I(b), RAST(c), RRST(d), RIGT(e), PSO(f) and convergence curve (g), error bar (h) produced by the path planners.
ZENG et al.: INFORMATION-DRIVEN PATH PLANNING FOR HYBRID AERIAL UNDERWATER VEHICLES

Fig. 9. Scenario 5: Informative path generated by RAST*-J/E (a), RAST*-I(b), RAST(c), RRST(d), RIGT(e), PSO(f) and convergence curve (g), error bar (h) produced by the path planners.
reducing the number of iterations. In summary, for this scenario, if a high-stability algorithm is needed, this article recommends the RAST∗-I/E algorithm; if a fast convergence algorithm is required, this article recommends the PSO algorithm.

### D. Robustness Analysis

To verify the robustness of these six algorithms, this section will conduct simulation experiments to evaluate three scenarios from single constraint conditions, double constraint conditions, and the sea–air environment with different weights, respectively. Each scenario is simulated 100 times, i.e., 100 different maps of sea–air environments and velocity fields are randomly generated, and the task start and end positions are randomly selected. The specific settings of the three mission scenarios are as follows.

**Scenario 1:** The HAUV carries a limited energy $E_{\text{max}}$, but has no limits for the mission time.

**Scenario 2:** The HAUV carries a limited energy $E_{\text{max}}$ with a mission time of a random number in the interval $[1 \text{ h}, 3 \text{ h}]$.

**Scenario 3:** The HAUV carries a limited energy $E_{\text{max}}$ with a mission time of $T_{\text{max}} = 3 \text{ h}$ and atmospheric and oceanic information weights $\kappa_{\text{air}}$ and $\kappa_{\text{sea}}$ are random integers in the interval $[1, 5]$.

The robustness experiments are judged by which algorithm can collect more information with the same setting. The algorithm with the highest number of winning results among 100 randomized simulations for each scenario is considered the most robust and more suitable for practical applications in such scenarios.

The results are shown in Fig. 10. The RAST∗-I/E algorithm wins over the other five algorithms by an absolute margin, indicating that the paths optimized by the RAST∗-I/E algorithm are more likely to capture more information, confirming the superiority of the RAST∗-I/E algorithm in solving the HAUV information-driven path-planning problem. On the other hand, as an excellent classical global optimization algorithm, the PSO algorithm is slightly less robust than the RAST∗-I/E algorithm proposed in this article in terms of adaptability and optimization capability.

### VI Summary

This article presents a new RAST∗-I/E algorithm for information-driven path-planning problems of HAUV. This RAST∗-I/E algorithm innovatively combines the sampling strategy based on the tournament point selection method, information heuristic search process, and the framework of the RRT∗ algorithm. To compare the effectiveness of the newly designed structure in the RAST∗-I/E algorithm, the RAST∗-I algorithm with the heuristic factor of total information is designed according to the information heuristic search process; the RRST∗ algorithm based on the random sampling strategy is designed according to the sampling strategy; and the RAST without the information heuristic search and parent node reshaping process is designed according to the presence or absence of this process algorithm without this process according to the presence or absence of information heuristic search and parent node reshaping process. Moreover, the classical RIGT algorithm and the PSO algorithm are designed as comparison algorithms. The simulation experiments were conducted to compare the above six algorithms’ optimization performance, speed, and stability through five cases. The sampling strategy of the RAST∗-I/E algorithm based on the tournament point selection method guides the adaptive sampling tree to explore the regions where information with higher values are located. The information heuristic search process is the key to preventing the algorithm from falling into local optimal paths. The RAST∗-I/E algorithm combines the advantages of tournament point selection, information heuristic...
search, and RRT* algorithm to efficiently search the air–sea environment. Therefore, the obtained sampling path can collect the most information, which ensures the accuracy and robustness of the algorithm.

REFERENCES

[1] L. R. Centurioni et al., “Global in situ observations of essential climate and ocean variables at the air-sea interface,” Front. Mar. Sci., vol. 6, 2019, Art. no. 419, doi: 10.3389/fmars.2019.00419.

[2] J. Elston, B. Argrow, M. Stachura, D. Weibel, D. Lawrence, and D. Pope, “Overview of small fixed-wing unmanned aircraft for meteorological sampling,” J. Atmos. Ocean. Technol., vol. 32, no. 1, pp. 97–115, 2015, doi: 10.1175/JTECH-D-13-00236.1.

[3] J. Meng, Y. Liu, R. Bucknall, W. Guo, and Z. Ji, “Anisotropic GPMP2: A fast continuous-time Gaussian process based motion planner for unmanned surface vehicles in environments with ocean currents,” IEEE Trans. Autom. Sci. Eng., vol. 19, no. 4, pp. 3914–3931, Oct. 2022, doi: 10.1109/TASE.2021.3139163.

[4] Z. Zeng, L. Lian, K. Sammut, F. He, Y. Tang, and A. Lammas, “A survey on path planning for persistent autonomy of autonomous underwater vehicles,” Ocean Eng., vol. 110, pp. 303–313, 2015, doi: 10.1016/j.oceaneng.2015.10.007.

[5] M. Yang, Y. Wang, Y. Liang, and C. Wang, “A new approach to system design optimization of underwater gliders,” IEEE/ASME Trans. Mechatron., vol. 27, no. 5, pp. 3494–3505, Oct. 2022, doi: 10.1109/TMECH.2022.3143125.

[6] Z. Zeng, K. Sammut, L. Lian, A. Lammas, F. He, and Y. Tang, “Rendezvous path planning for multiple autonomous marine vehicles,” IEEE J. Ocean. Eng., vol. 43, no. 3, pp. 640–664, Jul. 2018, doi: 10.1109/JOE.2017.2723058.

[7] C. Lyu et al., “Toward a gliding hybrid aerial underwater vehicle: Design, fabrication, and experiments,” J. Field Robot., vol. 39, no. 5, pp. 543–556, 2022, doi: 10.1002/rob.22063.

[8] D. Lu et al., “Design, fabrication, and characterization of a multimodal hybrid aerial underwater vehicle,” Ocean Eng., vol. 219, 2021, Art. no. 108324, doi: 10.1016/j.oceaneng.2020.108324.

[9] X. Liang, C. Liu, and Z. Zeng, “Multi-domain informative coverage path planning for a hybrid aerial underwater vehicle in dynamic environments,” Machines, vol. 9, no. 11, 2021, Art. no. 278, doi: 10.3390/machines9110278.

[10] Z. Zeng, C. Lyu, Y. Bi, Y. Jin, D. Lu, and L. Lian, “Review of hybrid aerial underwater vehicle: Cross-domain mobility and transitions control,” Ocean Eng., vol. 248, 2022, Art. no. 110840, doi: 10.1016/j.oceaneng.2022.110840.

[11] R. Hu et al., “Modeling, characterization and control of a piston-driven buoyancy system for a hybrid aerial underwater vehicle,” Appl. Ocean Res., vol. 120, 2022, Art. no. 102925, doi: 10.1016/j.apor.2021.102925.

[12] A. Vasiljević, B. Nad, F. Mandić, N. Mišković, and Z. Vukić, “Coordinated navigation of surface and underwater marine robotic vehicles for ocean sampling and environmental monitoring,” IEEE/ASME Trans. Mechatron., vol. 22, no. 3, pp. 1174–1184, Jun. 2017, doi: 10.1109/TMECH.2017.2684423.

[13] J. Binney, A. Krause, and G. S. Sukhatme, “Informative path planning for an autonomous underwater vehicle,” in Proc. IEEE Int. Conf. Robot. Autom., 2010, pp. 4791–4796.
A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart, “Receding horizon “next-best-view” planner for 3D exploration,” in Proc. IEEE Int. Conf. Robot. Auton., 2016, pp. 1462–1468.

L. Zacchini, M. Franchi, and A. Ridolfi, “Sensor-driven autonomous underwater inspections: A receding-horizon RRT-based view planning solution for AUVs,” J. Field Robot., vol. 39, no. 5, pp. 499–527, 2022.

F. Colas, S. Mahesh, F. Pomereau, M. Liu, and R. Siegwart, “3D path planning and execution for search and rescue ground robots,” in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., 2013, pp. 722–727.

Y. Wang, M. Ramezani, and M. Fallon, “Actively mapping industrial structures with information gain-based planning on a quadruped robot,” in Proc. IEEE Int. Conf. Robot. Auton., 2020, pp. 8609–8615.

T. Somers and G. A. Hollinger, “Human–robot planning and learning for marine data collection,” Auton. Robots, vol. 40, no. 7, pp. 1123–1137, 2016.

T. O. Fossum et al., “Information-driven robotic sampling in the coastal ocean,” J. Field Robot., vol. 35, no. 7, pp. 1101–1121, 2018.

E. L. Johnson, G. L. Nemhauser, and M. W. Savelsbergh, “Progress in linear programming-based algorithms for integer programming: An exposition,” Inform. J. Comput., vol. 12, no. 1, pp. 2–23, 2000.

N. K. Yilmaz, C. Evangelinos, P. F. Lermusiaux, and N. M. Patrikalakis, “Path planning of autonomous underwater vehicles for adaptive sampling using mixed integer linear programming,” IEEE J. Ocean. Eng., vol. 33, no. 4, pp. 522–537, Oct. 2008.

A. Singh, A. Krause, C. Guestrin, and W. J. Kaiser, “Efficient informative sensing using multiple robots,” J. Artif. Intell. Res., vol. 34, pp. 707–755, 2009.

J. Binney and G. S. Sukhatme, “Branch and bound for informative path planning,” in Proc. IEEE Int. Conf. Robot. Autom., 2012, pp. 2147–2154.

P. Sankiewicz, Y. T. Tan, and M. Kobilarov, “Adaptive sampling with an autonomous underwater vehicle in static marine environments,” J. Field Robot., vol. 38, no. 4, pp. 572–597, 2021.

K. D. Heaney, G. Gawarkiewicz, T. F. Duda, and P. F. Lermusiaux, “Nonlinear optimization of autonomous undersea vehicle sampling strategies for oceanographic data-assimilation,” J. Field Robot., vol. 24, no. 6, pp. 437–448, 2007.

K. D. Heaney, P. F. Lermusiaux, T. F. Duda, and P. J. Haley, “Validation of genetic algorithm-based optimal sampling for ocean data assimilation,” Ocean Dyn., vol. 66, no. 10, pp. 1209–1229, 2016.

M. Armazendia, D. Gregor, D. G. Reina, and S. L. Toral, “An evolutionary approach to constrained path planning of an autonomous surface vehicle for maximizing the covered area of Ypacarai Lake,” Soft Comput., vol. 23, no. 5, pp. 1723–1734, 2019.

S. Frolow, B. Garau, and J. Bellingham, “Can we do better than the grid survey: Optimal synoptic surveys in presence of variable uncertainty and decorrelation scales,” J. Geophys. Res., Oceans, vol. 119, no. 8, pp. 5071–5090, 2014.

H. Zhou, Z. Zeng, and L. Lian, “Adaptive re-planning of AUVs for environmental sampling missions: A fuzzy decision support system based on multi-objective particle swarm optimization,” Int. J. Fuzzy Syst., vol. 20, no. 2, pp. 650–671, 2018.

C. Xiong, D. Lu, Z. Zeng, L. Lian, and C. Yu, “Path planning of multiple unmanned marine vehicles for adaptive ocean sampling using elite group-based evolutionary algorithms,” J. Intell. Robot. Syst., vol. 99, no. 3, pp. 875–889, 2020.

G. Colmenares, F. Halal, and M. B. Zaremba, “Ant colony optimization for data acquisition mission planning,” Manage. Prod. Eng. Rev., vol. 5, pp. 3–11, 2014.

C. Xiong, Z. Zeng, and L. Lian, “Path planning of multi-modal underwater vehicle for adaptive sampling using Delaunay spatial partition-ant colony optimization,” in Proc. OCEANS’13 MTS/IEEE Kobe Techno-Oceans, 2013, pp. 1–8.

Y. Hu, D. Wang, J. Li, Y. Wang, and H. Shen, “Adaptive environmental sampling for underwater vehicles based on ant colony optimization algorithm,” in Proc. Glob. Ocean., 2020, pp. 1–9.

R. P. Anderson, G. S. Dinolfo, D. Milutinović, and A. M. Moore, “Maximally-informative regional ocean modeling system (ROMS) navigation of an AUV in uncertain ocean currents,” in Proc. ASME 5th Dyn. Syst. Control. Conf., 2012, vol. 45318, pp. 291–297.

G. Ferri, M. Cococcioni, and A. Alvarez, “Mission planning and decision support for underwater glider networks: A sampling on-demand approach,” Sensors, vol. 16, no. 1, 2016, Art. no. 28.
Zheng Zeng (Senior Member, IEEE) received the B.Eng. degree in electrical and electronic engineering from Hunan University, Changsha, China, in 2010, and the Ph.D. degree in ocean engineering from Flinders University, Adelaide, SA, Australia, in 2015.

He is currently an Associate Research Professor with Shanghai Jiao Tong University, Shanghai, China. His principal areas of expertise lie in autonomous marine vehicles, including optimal guidance, navigation, and control systems. He is the Project Leader for multiple projects funded by Shanghai Sailing Program, Natural Science Foundation of Shanghai and National Natural Science Foundation of China.

Dr. Zeng is a Reviewer for multiple international scientific journals and a committee member of the IEEE-OES/MTS Ocean’s 16 Shanghai conference. He is currently the Secretary of the IEEE-OES Shanghai Section.

Chengke Xiong received the B.Eng. degree in naval architecture and ocean engineering from Huazhong University of Science and Technology, Wuhan, China, in 2015, and the Ph.D. degree from the School of Naval Architecture, Ocean & Civil Engineering, Shanghai Jiao Tong University, Shanghai, China, in 2021. His Ph.D. works are focused on optimization algorithms, adaptive ocean sampling, and path planning of unmanned marine vehicles.

Xinyi Yuan received the B.Eng. degree in marine technology from Dalian University of Technology, Dalian, China, in 2022. She is currently working toward the M.Sc. degree in marine science with the School of Oceanography, Shanghai Jiao Tong University, Shanghai, China. Her research interests include underwater vehicles, hybrid aerial underwater vehicles, path planning, and adaptive sampling.

Hexiong Zhou received the B.Eng. degree in electrical engineering and automation from Northwestern Polytechnical University, Xi’an, China, in 2015, the M.Eng. degree in control engineering from Harbin Engineering University, Harbin, China, in 2018, and the Ph.D. degree from the School of Naval Architecture, Ocean & Civil Engineering, Shanghai Jiao Tong University, Shanghai, China, in 2021. He is currently a Postdoctoral Fellow with the Institute of Oceanography, Shanghai Jiao Tong University. His research interests include multimodal underwater vehicles, hybrid aerial underwater vehicles, path planning, and adaptive robust control.

Yuling Bai received the B.Eng. degree in marine science from Shandong University, Weihai, China, in 2020, and the M.Sc. degree in marine science from the School of Oceanography, Marine Technology, Shanghai Jiao Tong University, Shanghai, China, in 2022. His research interests include floats, hybrid aerial underwater vehicles, and embedded design.

Yufei Jin received the M.Eng. degree in advanced aeronautical engineering from Imperial College London, London, U.K., in 2019. He is currently working toward the Ph.D. degree from the School of Oceanography, Shanghai Jiao Tong University, Shanghai, China.

His current research focuses on the system design of hybrid aerial underwater vehicles.

Di Lu received the B.Eng. degree in naval architecture and ocean engineering from Huazhong University of Science and Technology, Wuhan, China, in 2015, and the Ph.D. degree from the School of Naval Architecture, Ocean & Civil Engineering, Shanghai Jiao Tong University, Shanghai, China, in 2021. His research interests include underwater vehicles, hybrid aerial underwater vehicles, nonlinear time-varying systems, adaptive control, and backstepping control.

Lian Lian (Senior Member, IEEE) received the B.S. and M.S. degrees in naval architecture and ocean engineering and the Ph.D. degree in technology management from Shanghai Jiao Tong University, Shanghai, China, in 1982, 1985, and 2012, respectively. Since 1998, she has been a Full Professor with Shanghai Jiao Tong University. From 1999 to 2011, she was a member of the Expert Group of National Hi-Tech Program (863 Program). From 2008 to 2011, she was a member of the Expert Group of the National Science Foundation of China. She is currently the Vice Dean of the Institute of Oceanography, Shanghai Jiao Tong University and a member of the IEEE/OES Administrative Committee and the IEEE/OES Shanghai Chapter Chair. She has been in charge of eight projects as PI/Chief Designer with total funding of more than 90 million RMB, which include national and municipal research projects funded by the National Hi-Tech 863 Program, National Key Project, National Natural Science Foundation of China, National R&D Program of Marine Technology, and the R&D Program of Science and Technology Committee of Shanghai Municipal Government. Her research mainly focuses on underwater vehicles and marine observation.