

Article

A Hierarchical Planning Method for AUV Search Tasks Based on the Snake Optimization Algorithm

Zhiwen Wen ¹, Zhong Wang ¹, Xiangdong Wen ², Chenxi Niu ², Pei Wang ^{2,*} and Daming Zhou ²

¹ Xi'an Precision Machinery Research Institute, Xi'an 710077, China; wenzhiwen0506@163.com (Z.W.)

² School of Astronautics, Northwestern Polytechnical University, Xi'an 710072, China

* Correspondence: jimmy5579@nwpu.edu.cn

Abstract: In a complex and dynamic battlefield environment, enabling autonomous underwater vehicles (AUVs) to reach dynamic targets in the shortest possible time using global autonomous planning is a key issue affecting the completion of search tasks. In this study, a hierarchical AUV task planning method that uses a combination of hierarchical programming and a snake optimization algorithm is proposed for two typical cases where the platform can provide initial target information. This method decomposes the search task problem into a three-level programming problem, with the outer task planning goal of achieving the shortest encounter time between AUV and dynamic targets; the goal of task planning in the middle layer is to achieve the shortest actual navigation time for AUVs under different operating conditions; and the internal task planning is responsible for considering the comprehensive trajectory optimization under navigation constraints such as threat zone, path length, and path smoothness. The snake optimization algorithm was used for solving each layer of task planning. The feasibility of the proposed method was verified through simulation experiments of AUV search tasks under two types of initial target information conditions. The simulation results show that this method can achieve task planning for AUV searching for dynamic targets under various constraint conditions, optimize the encounter time between AUV and dynamic targets, and have strong engineering practical value. It has certain reference significance for task planning problems similar to underwater unmanned equipment.

Keywords: autonomous underwater vehicles; search tasks; hierarchical programming approach; snake optimization algorithm



Citation: Wen, Z.; Wang, Z.; Wen, X.; Niu, C.; Wang, P.; Zhou, D. A Hierarchical Planning Method for AUV Search Tasks Based on the Snake Optimization Algorithm. *Sensors* **2024**, *24*, 7196. <https://doi.org/10.3390/s24227196>

Academic Editor: Sergio Toral Marin

Received: 16 September 2024

Revised: 22 October 2024

Accepted: 24 October 2024

Published: 10 November 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

In recent years, with the rapid development of science and technology and the deepening understanding of the ocean, underwater space has become a new focus of international strategic competition. As a multiplier of naval power, the autonomous underwater vehicle (AUV) has seen unprecedented development in applications such as underwater reconnaissance, underwater communication, anti-submarine and anti-mine warfare, and information warfare. As a manifestation of AUV's intelligence level, the capability of autonomous task planning significantly affects the success rate of AUV missions. Current global task planning methods mainly focus on solving planning problems for static environments or known navigation endpoints. However, in practical applications, due to the partial observability of the task environment and the dynamic changes in target information, the environmental information relied on for planning is uncertain. In fact, AUVs are more often used for searching and exploring dynamic targets. Before being launched by the platform, task planning can be divided into two categories based on the completeness of the initial dynamic target information obtained through platform detection: task planning under initially complete target information and task planning under initially incomplete target information. Initial complete target information refers to the relatively complete dynamic target information that can be obtained by the platform before deploying the AUV,

including target location, distance, heading, and speed information. In contrast, initial incomplete target information refers to the situation where the platform can only obtain approximate dynamic target information before deploying the AUV, such as target location and approximate distance information. In light of these circumstances, the key issue that urgently needs to be solved is how to efficiently perform global task planning for the AUV based on limited platform target information before deploying the AUV, so that the AUV has a high degree of autonomy for locating dynamic targets.

The United States is the earliest country to research and equip combat aircraft mission planning systems and has experienced three generations of development [1,2]. Greshler et al. [3] introduced cooperative conflict-based search (Co-CBS) to integrate the CBS algorithm based on a multi-agent collaboration framework, enabling collaborative task allocation and path planning for multiple robots. Other studies proposed a modified A-Star algorithm containing the bidirectional sector expansion and variable step search strategy for penetration path planning for stealth unmanned aerial vehicles (SUAVs) [4]. An innovative path planning algorithm that synergizes the A* algorithm with the rapidly-exploring random tree (RRT) approach was proposed to improve planning efficiency and path smoothness [5]. However, the above-mentioned path planning algorithm still has the disadvantage of easily falling into the local optimal solution.

To improve the efficiency of the solution, some scholars have utilized intelligence heuristic algorithms for task allocation, such as genetic algorithms [6], particle swarm optimization [7], and simulated annealing (SA) algorithms [8]. Yu Jing et al. [9] designed a double-layer mutually coupled task planning solution strategy that divides the task planning model into upper-level task allocation and lower-level task sequence optimization, with specific optimization methods and steps for each layer.

In order to break through the inherent scalability limitations of traditional centralized control path planning algorithms, Guo et al. [10] designed a decentralized path planning algorithm predicated on deep reinforcement learning to enhance the flexibility, robustness, and scalability of automated guided vehicles (AGVs). Chen et al. [11] and Xiao et al. [12] described the task planning problem as a mixed-integer linear programming (MILP) model and solved it using a linear programming solver, which can yield stable and effective optimization results. In the literature [13], the strike task is decomposed into a multi-objective task allocation problem under time constraints and a trajectory planning problem, with the two sub-problems being mutually coupled. Wang Bo et al. [14] decomposed the planning task into an upper-level planning for designing intermediate configurations and a lower-level planning for solving the movement methods of achieving those intermediate configurations. Each level of planning is solved independently, reducing the uncertainty and complexity of the satellite reconstruction planning problem. Bai et al. [15] proposed a novel two-layer algorithm SA-reCBS that cascades the simulated annealing algorithm and conflict-based search to solve multi-robot task assignment and path-finding problem without requiring a pre-bundle of tasks to groups with the same number of groups as the number of robots. Yang Han and Yu et al. [16] proposed a heuristic called Space Utilization Optimization (SUO) for path planning. This heuristic aims to significantly reduce conflicts among robots while maintaining the optimality of the paths. Chen Zihao et al. [17] proposed a cluster task planning method based on a double-layer optimization model. The lower-level optimization focuses on trajectory planning, while the upper-level optimization is task allocation, improving the optimality of the task planning results.

From top-level planning to layered and gradual implementation, this approach helps to reduce the complexity of the problem. In terms of task planning solutions, heuristic algorithms such as particle swarm optimization, differential evolution, wolf pack algorithm [18], genetic algorithms [19,20], ant colony optimization [21], and firefly algorithm [22] have received widespread attention from scholars in recent years due to their simplicity and effectiveness. Zhu et al. [23] introduced a biologically inspired integrated self-organizing map algorithm for task allocation and path planning of AUV systems in three-dimensional underwater obstacle avoidance environments. Zhang et al. [24] used the sine cosine algo-

rithm to improve the Harris Hawk algorithm for path planning, enhancing its robustness. However, they were unable to estimate the convergence speed of the algorithm in complex environments. Yu et al. [25] proposed a traversal multi-target path planning method for multiple unmanned surface vehicles (USVs). The proposed algorithm consists of two parts: First, the improved Hungary algorithm is proposed for multi-target task allocation. Second, the virtual field sampling algorithm and the ocean current constraint function are introduced in the RRT* algorithm for path planning. Through the improvement of the algorithm, the efficiency of multi-target task allocation improves, and the time spent path planning is reduced. However, the algorithm's slow convergence accuracy remains a problem. Han et al. [26] designed a full-dimensional ABC algorithm with an adaptive factor to enhance the search capability of AUVs. Although the improved algorithm solves the problem of planning global paths for AUVs to search for targets, it easily falls into the local optimum. Zou et al. [27] proposed a novel population-based iterated greedy (PIG) algorithm for multi-automated guided vehicles (multi-AGV) scheduling problems involving unloading safety detection. In the PIG, a hyper-heuristic based on neighborhood operators and a population-based initialization method are proposed to obtain multiple high-quality initial solutions. In another study, Han et al. [28] propose a novel mixed integer linear programming (MILP) model and a dual population collaborative genetic algorithm (DCGA). The integration problems of the flexible job shop scheduling problem (FJSP) and minimizing the makespan of AGVs were studied. Cui et al. [29] proposed a mixed-integer linear programming model and a discrete Jaya (DJaya) algorithm to the multi-AGVs scheduling problem with dynamic unloading time. This algorithm consists of two parts: (1) A heuristic based on the ant colony algorithm is used in the initial solution; (2) two DJaya operators are designed for the best and worst solutions. The computational results show that the proposed DJaya algorithm is superior to the existing algorithms in tackling the considered problem.

Although the above modifications have improved efficiency, they cannot predict the convergence speed of algorithms in complex environments, and there is an imbalance between global exploration and local development. The above algorithms have higher optimization accuracy than traditional classical algorithms and show significant improvement in path planning and task allocation in unmanned systems. However, this does not necessarily mean that these algorithms have the best optimization accuracy and task planning quality. Their optimization accuracy and path planning quality in different scenarios still need to be improved. Attempting to improve different algorithms through different strategies for solving task planning problems in unmanned systems still has research significance.

The task planning problem for AUVs to track dynamic targets can be decomposed into two coupled sub-problems: the problem of being able to intercept the dynamic target and the problem of minimizing the time to interception. The aim of this paper is to propose an autonomous global task planning method for AUVs by combining two typical types of initial observed target information.

This study proposes a hierarchical task planning method based on the snake algorithm, which hierarchically designs the task allocation problem of AUVs. A multi-layer task planning model is proposed for use in conjunction with the snake optimization algorithm to achieve multi-layer nested optimization and address the task planning problem of AUVs under typical target initial information conditions. Using MATLAB software, this study develops algorithm validation programs, constructs complex underwater task scenarios, sets initial scenarios for typical tasks, uses the designed algorithms for task planning, and verifies its application effectiveness.

The main innovations of this study are as follows: (1) The adoption of a three-level planning strategy to address the aforementioned problems, which makes it easier to obtain global optimal solutions for dynamic targets; (2) the introduction of a snake optimization algorithm into each level of task planning, enabling better objective function values; (3) the development of two different three-level task planning methods tailored to the

two maintypes of initial target detection information provided by the platform, effectively addressing practical engineering problems.

This paper is organized as follows: Section 2 describes the snake optimization algorithm and hierarchical task planning method. Section 3 gives the experimental design and discusses the experimental results. Finally, Section 4 provides our conclusion.

2. Materials and Methods

2.1. Assumptions

For the purpose of this study, which focuses on AUV autonomous task planning, and in consideration of the practical working characteristics of AUVs, the following simplifications and assumptions are made:

1. The AUV is treated as a point mass within the vast underwater space.
2. Obstacles and threat areas are considered collision zones, and irregular collision zones are uniformly “inflated” to cylindrical shapes for simplicity.
3. Factors such as tidal currents, sea currents, electronic interference, and other disturbances are not considered.
4. During simulations, both the AUV and the dynamic target move with constant speeds.

2.2. Environment Modeling

The underwater three-dimensional space is modeled using a grid-based approach. This study considers a scenario where an underwater platform deploys an AUV to search for and monitor a dynamic target. After the underwater platform acquires some initial target information, it releases the AUV to conduct a close-range search and surveillance of the dynamic target.

In Figure 1, we assume that each obstacle or threat is defined within a cylinder with a projection that has a center coordinate $C_k = [x, y, z]$ and a radius R_k . R_k is the maximum distance between the surface and the center of an irregular sphere. D_k represents the AUV safety margin distance and S is the critical distance from the AUV to the collision zone, which depends on the specific application, operating environment, and positioning accuracy. In this simulation, R is the radius of the cylinder, which is identified as an obstacle or threat and is defined as $R = R_k + D_k$. Here, based on the size of the AUV, select $D_k = 20$ m. x and y represent the center coordinates of the obstacle, while z represents its height. In practical applications, the terrain height needs to be added to z for processing. The obstacle data used in the simulation are presented in Table 1.

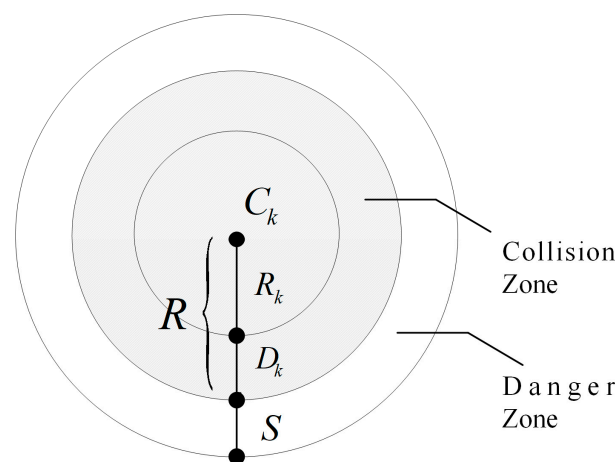
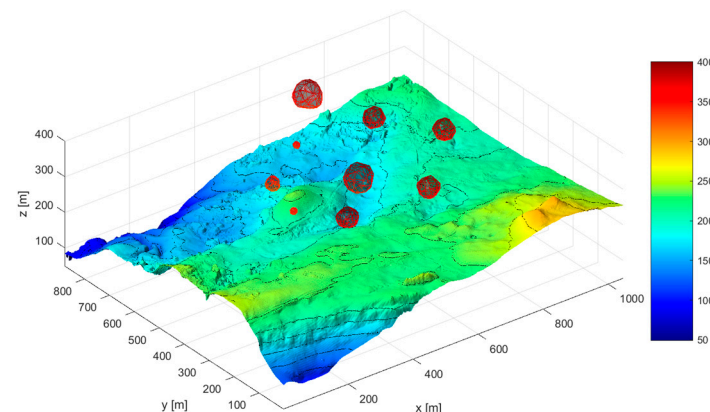


Figure 1. Obstacle symbol description.

Table 1. Obstacle parameters.

x (m)	y (m)	z (m)	R (m)
350	500	100	60
600	200	150	70
500	350	150	80
350	200	150	70
700	550	150	70
650	750	150	80
800	400	150	70
300	350	100	50
500	600	100	50

The environment design program was developed in MATLAB 2018b. We obtained seabed elevation information by loading elevation image files in tif format, generating seabed terrain grid coordinates using the meshgrid command, and establishing the seabed terrain. We set the obstacle position in Matlab, using the meshgrid() command to generate the surface coordinates of the obstacle, using the surf() command to smooth and generate irregular obstacles, and taking the maximum distance from the center as the radius of the obstacle. After displaying obstacles on the underwater terrain image, we completed the environment modeling. The environment model is shown in Figure 2. Several obstacles of different sizes and radii were added to the constructed seabed model to simulate the threat areas that the AUV may encounter during navigation.

**Figure 2.** Submarine model after adding obstacles.

2.3. Snake Optimization Algorithm

Inspired by the unique mating behavior of snakes, Fatma A. Hashim and Abdelazim G. Hussien proposed the snake optimizer (SO) algorithm in 2022 [30]. The SO algorithm is a new meta-heuristic optimization algorithm that simulates the different behavioral patterns of snakes under different temperatures and food conditions to search for the optimal value. The flow of the SO algorithm is shown in Figure 3. The definitions of the parameters involved in the SO algorithm are shown in Table 2.

Table 2. SO Algorithm Parameter Definition.

Symbol	Meaning
Am, Af	Represent the ability of males and females to find things, respectively
rand	Represents a random number within the range [0,1]
N, Nm, Nf	Represent the snake population size, the number of males, and the number of females, respectively

Table 2. Cont.

Symbol	Meaning
t, T	Represent the current iteration number and the maximum number of iterations, respectively
$c1, c2$ and $c3$	Are all constants, respectively, 0.5, 0.05, and 2
Q	Food quantity
Temp	Temperature
FF, FM	Represent the fighting ability of males and females, respectively
$X_{i,max}, X_{i,min}$	Solve the upper and lower boundaries of the problem
Limits	Solve the difference between the upper and lower boundaries of the problem, that is, $X_{max}-X_{min}$
$X_{i,m}, X_{i,f}, X_i$	Represent the position of the i th individual in males, females, and the entire population, respectively
$f_{i,m}, f_{i,f}, f_i$	Represent the fitness of the i th individual in males, females, and the entire population, respectively
$X_{best,m}, X_{best,f}$ and X_{food}	Represent the position of the best individual in males, females, and the entire population, respectively
$f_{best,m}, f_{best,f}$ and f_{food}	Represent the fitness of the best individual in males, females, and the entire population, respectively
$X_{rand,m}, X_{rand,f}$	Random positions of males and females
$f_{rand,m}, f_{rand,f}$	Random positions of males and fitness of females
$X_{worst,m}, X_{worst,f}$	Represent the position of the worst individual in males and females, respectively.

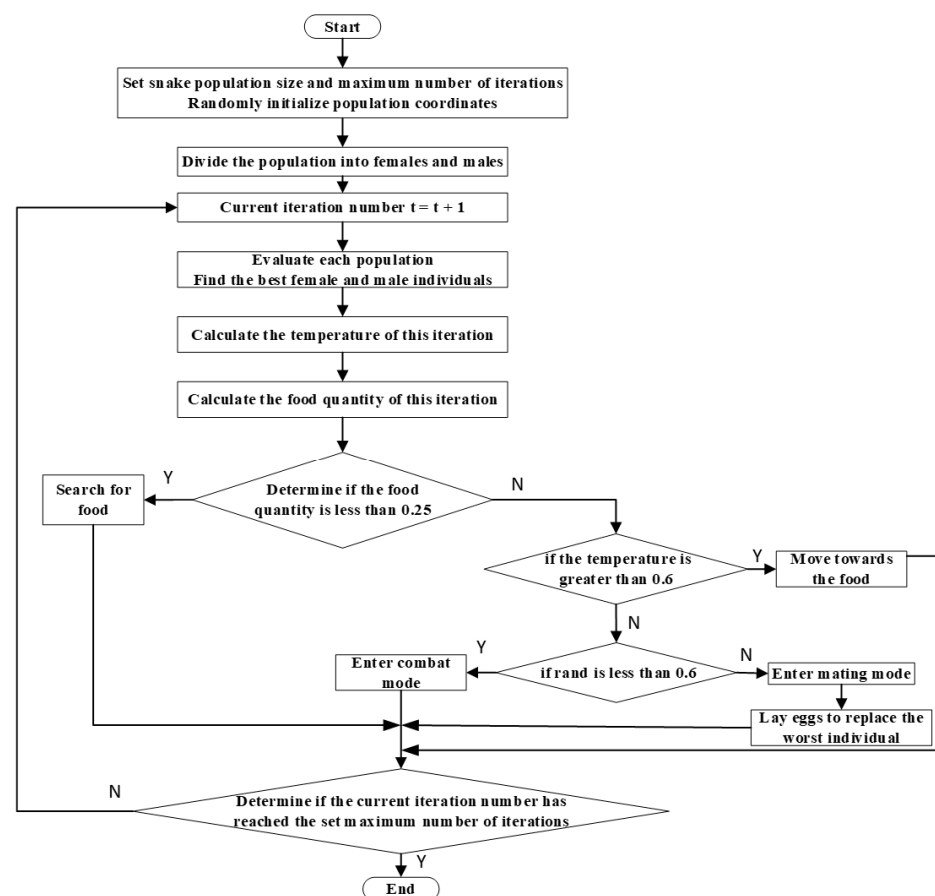


Figure 3. SO algorithm flowchart. (“N” = “No”, “Y” = “Yes”, “t” = “iteration number”, subsequent settings are the same here).

First, the snake population is initialized:

$$X_i = X_{i,min} + \text{rand} \times (X_{i,max} - X_{i,min}), \quad (1)$$

where X_i represents the position of the i th individual, rand is a random number between 0 and 1, and $X_{i,min}$ and $X_{i,max}$ are the lower and upper bounds of the problem, respectively.

After the snake population is randomly initialized in the search space, the population is evenly divided into two groups: male and female. The number of males and females is represented by the following two formulas, respectively:

$$\begin{aligned} N_m &\approx \frac{N}{2} \\ N_f &= N - N_m \end{aligned} \quad (2)$$

where N represents the total population size, N_m is the number of male individuals, and N_f is the number of female individuals. From each group, the best individual is identified to obtain the positions of the best male, best female, and food, denoted as $f_{best,m}$, $f_{best,f}$, and f_{food} , respectively. The definition of temperature is as follows:

$$\text{Temp} = \exp\left(\frac{-t}{T}\right), \quad (3)$$

where t represents the current iteration number and T represents the maximum iteration number. The amount of food is defined as:

$$Q = c_1 \times \exp\left(\frac{t-T}{T}\right), \quad (4)$$

where c_1 is a constant, taken as 0.5.

The exploration phase describes the environmental factors of cold regions and food, where snakes do not actively search for food in their immediate surroundings. If $Q < \text{Threshold}$ (where $\text{Threshold} = 0.25$), snake individuals search for food by randomly selecting positions and updating their own positions accordingly. The exploration phase of the snake population is simulated as follows:

$$X_{i,m(f)}(t+1) = X_{\text{rand},m(f)}(t) \pm c_2 \times A_{m(f)} \times ((X_{\max} - X_{\min}) \times \text{rand}) + X_{\max}, \quad (5)$$

In the formula: $X_{i,m(f)}$ represents a male or female individual, rand is a random number in the range $[0,1]$; c_2 is taken as 0.05; and $A_{m(f)}$ represents the ability of males or females to search for food, which is given by the following formula:

$$A_{m(f)} = \exp\left(\frac{-f_{\text{rand},m(f)}}{f_{i,m(f)}}\right), \quad (6)$$

where $f_{\text{rand},m(f)}$ is the fitness value of the randomly selected male or female individual position $X_{\text{rand},m(f)}$, and $f_{i,m(f)}$ is the fitness value of the i th individual position $X_{i,m(f)}$ in the male or female population.

During the exploitation phase, there is food available, i.e., $Q > \text{Threshold}$. If the temperature is high, represented by $\text{temperature} > \text{Threshold}$ (0.6), snakes will only search for food, and the position update formula is as follows:

$$X_{i,j}(t+1) = X_{\text{food}} \pm c_3 \times \text{Temp} \times \text{rand} \times (X_{\text{food}} - X_{i,j}(t)), \quad (7)$$

where $X_{i,j}(t)$ represents the individual's position and the best position for a snake individual is X_{food} , with c_3 taking the value of 2.

If the temperature is below the threshold, i.e., $temperature < Thresgold$, indicating a cold environment, individual snake will enter either combat or mating mode. The position update formula in combat mode is as follows:

$$X_{i,m(f)}(t+1) = X_{i,m(f)}(t) \pm c_3 \times FM(F) \times rand \times (Q \times X_{best,f(m)} - X_{i,m(f)}(t)) \quad (8)$$

In the formula, $X_{best,f(m)}$ represents the best position in the female (male) group, and $FM(F)$ represents the fighting ability of males or females. It is calculated as follows:

$$FM(F) = \exp\left(\frac{-f_{best,f(m)}}{f_i}\right), \quad (9)$$

In the formula, $f_{best,f(m)}$ represents the fitness value of the best male or female individual, while the current f_i is the fitness value of a snake individual.

In mating mode, simply replace $X_{best,f(m)}$ with $X_{i,f(m)}$, which represents the position of the i th female or male individual. Additionally, replace $FM(F)$ with $M_{m(f)}$ to represent the mating ability of males and females. It can be calculated as follows:

$$M_{m(f)} = \exp\left(\frac{-f_{i,f(m)}}{f_{i,m(f)}}\right), \quad (10)$$

If snake eggs hatch, the male and female with the worst fitness are selected to replace their positions, which can be represented as:

$$X_{worst,m(f)} = X_{min} + rand \times (X_{max} - X_{min}), \quad (11)$$

The pseudocode of the SO algorithm is in Algorithm 1.

Algorithm 1. Snake Optimization Algorithm

Input: pop, Iteration, fobj, dim, X_min, X_max
Output: Xfood, fval, gbest_t

1. Initialize ThresholdQ ThresholdT, c_1 , c_2 , c_3
2. $X = \text{initialization}(\text{pop}, \text{dim}, X_{\text{max}}, X_{\text{min}})$
3. Calculate fobj function fitness value for each X
4. $[GYbest, gbest] = \min(\text{fitness}), X_{\text{food}} = X(\text{gbest}, :)$
5. $Nm = \text{round}(\text{pop}/2)$, $Nf = \text{pop} - Nm$
6. $Xm = X(1:Nm, :)$, $Xf = X(Nm+1:\text{pop}, :)$
7. $\text{fitness}_m = \text{fitness}(1:Nm)$, $\text{bestX}_m = \text{bestX}(1:Nm, :)$ %Male snake population
8. $[\text{fitnessBest}_m, \text{gbest1}] = \min(\text{fitness}_m)$, $X_{\text{best}_m} = Xm(\text{gbest1}, :)$
9. $\text{fitness}_f = \text{fitness}(Nm+1:\text{pop})$, $\text{bestX}_f = \text{bestX}(Nm+1:\text{pop}, :)$ %Female snake population
10. $[\text{fitnessBest}_f, \text{gbest2}] = \min(\text{fitness}_f)$, $X_{\text{best}_f} = Xm(\text{gbest2}, :)$
11. for $i = 1:\text{Iteration}$
12. $\text{Temp} = \exp(-((t)/\text{Iteration}))$, $Q = c_1 \times \exp(((t - \text{Iteration})/\text{Iteration}))$
13. if $Q < \text{Threshold}$
14. update Xnewm use Xm by Equation (5)
15. update Xnewf use Xf by Equation (5)
16. else
17. if $\text{Temp} > \text{Thresold2}$
18. update Xnewm use Xbest by Equation (7)
19. update Xnewf use Xbest by Equation (7)
20. else
21. if $\text{rand} > 0.6$
22. update Xnewm use Xm by Equation (8)
23. update Xnewf use Xf by Equation (8)
24. else
25. update Xnewm use Xm by Equation (8) replace Fm with Mm

Algorithm 1. *Cont.*

```

26.  update Xnewf use Xf by Equation (8) replace Fm with Mm
27.  if rand > 0.5
28.    update Xnewm wrost by Equation (11)
29.    update Xnewf wrost by Equation (11)
30.  end
31.  end
32.  end
33.  end
34.  Calculate the objective function fitness value by Xnewm and update Xm
35.  Calculate the objective function fitness value by Xnewf and update Xf
36.  Update the optimal objective function fitness value and optimal male snake vector
37.  for the male snake population
38.  Update the optimal objective function fitness value and optimal female snake vector
39.  of the female snake population
40.  Update the global optimal objective function fitness value gbest_t
41.  and global optimal snake vector Xfood
42. end

```

2.4. Hierarchical Task Planning Method

Based on the level of completeness of the initial target information obtained by the platform, the AUV task planning can be typically divided into two categories: one is task planning under conditions of initially complete target information, and the other is task planning under conditions of initially incomplete target information.

2.4.1. Task Planning Under Initial Complete Goal Information

The schematic diagram of AUV planning under the condition of initially complete target information is shown in Figure 4. It indicates that the AUV has obtained relatively complete motion information about the target during the planning phase.

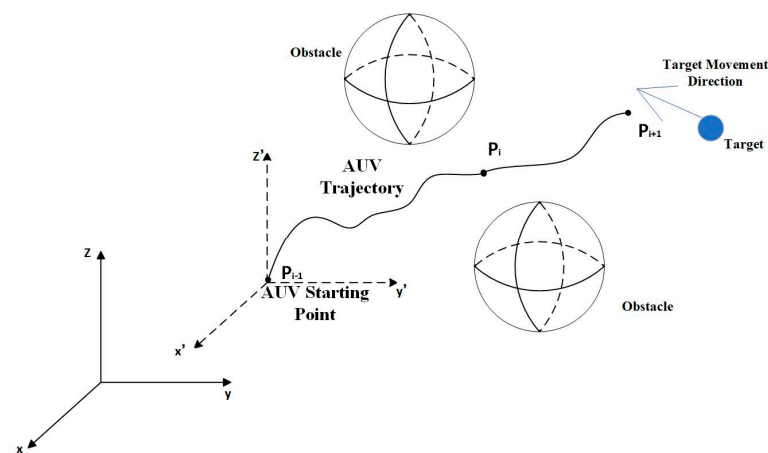


Figure 4. Schematic diagram of AUV planning under complete information conditions.

The overall design approach for task planning under complete information is shown in Figure 5. In this paper, a three-level optimized task planning based on the snake optimization algorithm is proposed. Given the target location, direction of motion, and speed, the task planning for the AUV aims to minimize the time taken for the AUV to encounter the target. The entire planning process is divided into three levels: external planning, intermediate planning, and internal planning.

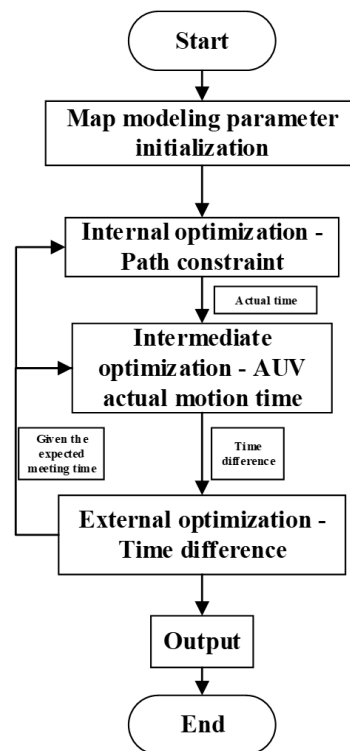


Figure 5. Overall idea diagram of three-layer task planning based on SO algorithm (Complete target information).

External planning focuses on optimizing the difference between the actual movement time of the AUV and the estimated time of encounter between the AUV and the target, which involves optimizing the overall objective function. Intermediate planning optimizes the actual movement time of the AUV. Internal planning optimizes the constraints associated with the AUV's path planning. By iteratively optimizing the AUV's actual movement time, intermediate planning aims to achieve the shortest possible time. Meanwhile, it calculates the time difference between the actual movement time and the given estimated time of encounter and feeds this information back to external planning for further optimization. The entire process iterates until an optimal result is achieved.

The objective of the three-level task planning based on the snake optimization algorithm is to minimize the time taken for the AUV to encounter the target, which means the difference between the actual movement time of the AUV and the estimated time of encounter should be as small as possible. Therefore, the objective optimization function of external planning is defined as:

$$\Delta t = |t_{auv} - t_{target}|, \quad (12)$$

where t_{auv} represents the actual movement time of the AUV; t_{target} is the given estimated-time for the encounter between the AUV and the target, which is also considered as the target movement time; and the difference between the actual movement time of the AUV and the estimated time of encounter is represented by Δt .

The objective of intermediate planning is to minimize the actual movement time of the AUV. The target optimization function is simply the practical movement time of the AUV, denoted as practical time.

Internal planning aims to optimize a better path for the AUV, which involves a multi-objective optimization process. The planning objectives include the optimal combination of path length, smoothness, and safety for the AUV.

The evaluation function for path length is the sum of the distances between each path node. The navigation path "Xi" is represented as a list of "n" waypoints that the

AUV needs to follow. Each waypoint corresponds to a path node in the search map, with coordinates $P_{ij} = (x_{ij}, y_{ij}, z_{ij})$. By representing the Euclidean distance between two nodes as $\|P_{ij}\vec{P}_{i,j+1}\|$, the cost F_1 associated with path length can be calculated as:

$$F_1(X_i) = \sum_{j=1}^{n-1} \|P_{ij}\vec{P}_{i,j+1}\|, \quad (13)$$

The evaluation function for path smoothness is a function of the path's turning angles, as shown in Figure 6.

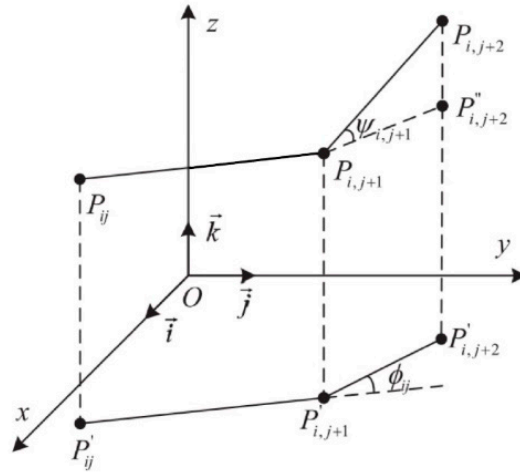


Figure 6. Schematic diagram for calculating turning and climbing angles.

The turning angle ϕ_{ij} is the angle between two consecutive path segments and the projections of these segments onto the horizontal Oxy plane, represented as vectors. Let $\vec{k}$ be the unit vector in the z -axis direction. The projection vectors can be calculated as:

$$P'_{ij}\vec{P}'_{i,j+1} = \vec{k} \times (P_{ij}\vec{P}_{i,j+1} \times \vec{k}), \quad (14)$$

Therefore, the turning angle is calculated as follows:

$$\phi_{ij} = \arctan\left(\frac{\|P'_{ij}\vec{P}'_{i,j+1} \times P'_{i,j+1}\vec{P}'_{i,j+2}\|}{P'_{ij}\vec{P}'_{i,j+1} \cdot P'_{i,j+1}\vec{P}'_{i,j+2}}\right), \quad (15)$$

The climbing angle ψ_{ij} is the angle between the path segment $P_{i,j}\vec{P}_{i,j+1}$ and its projection onto the horizontal plane $P'_{ij}\vec{P}'_{i,j+1}$. It is given by the following formula:

$$\psi_{ij} = \arctan\left(\frac{z_{i,j+1} - z_{ij}}{\|P'_{ij}\vec{P}'_{i,j+1}\|}\right), \quad (16)$$

The calculation of the smoothing cost is as follows:

$$F_2(X_i) = a_1 \sum_{j=1}^{n-2} \phi_{ij} + a_2 \sum_{j=1}^{n-1} |\psi_{ij} - \psi_{i,j-1}|, \quad (17)$$

where a_1 and a_2 are the penalty coefficients for the turning angle and climbing angle, respectively.

The planned path must meet safety conditions. Let K be the set of all collision zones, assuming that each collision zone has a center coordinate C_k and a radius R_k for its projection, as shown in Figure 7. The cost of a threat increases as the distance from the threat center decreases. For a given path segment $\|P_{ij} \vec{P}_{i,j+1}\|$, the associated threat cost is proportional to its distance d_k from C_k . By considering the safety margin distance D of the AUV and the hazardous distance S to the collision zone, the threat cost F_3 at waypoint P_{ij} for the obstacle set K is calculated as follows:

$$\begin{cases} F_3(X_i) = \sum_{j=1}^{n-1} \sum_{k=1}^K T_k(P_{ij} \vec{P}_{i,j+1}), \\ T_k(P_{ij} \vec{P}_{i,j+1}) = \begin{cases} 0, & \text{if } d_k > S + D + R_k \\ (S + D + R_k) - d_k, & \text{if } D + R_k < d_k \leq S + D + R_k \\ \infty, & \text{if } d_k \leq D + R_k. \end{cases} \end{cases} \quad (18)$$

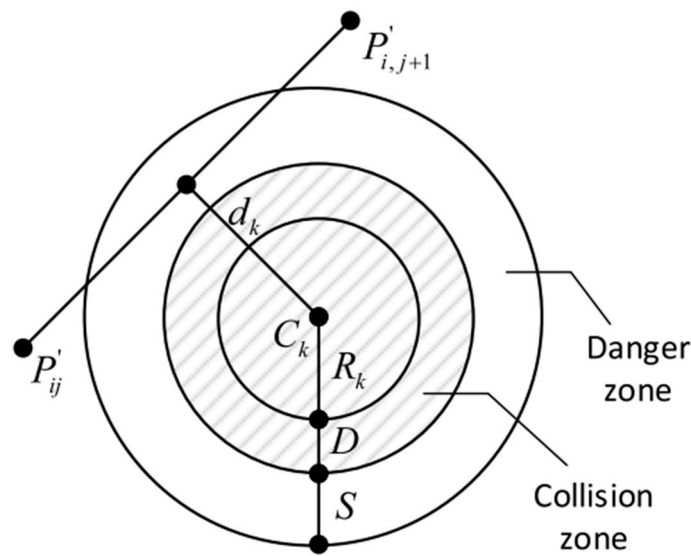


Figure 7. Determination of threat costs.

By considering the optimality, safety, and feasibility constraints associated with the path X_i , the overall cost function can be defined in the following form:

$$F(X_i) = \sum_{k=1}^3 b_k F_k(X_i), \quad (19)$$

where b_k is the weighting coefficient, and $F_1(X_i)$ to $F_3(X_i)$ are the costs associated with path length, path smoothness, and safety, respectively. The decision variable is X_i , which includes a list of n waypoints $P_{ij} = (x_{ij}, y_{ij}, z_{ij})$.

The detailed process of the three-level optimization task planning based on the snake optimization algorithm under initial complete information conditions is as follows:

1. Initialize parameters, providing the initial positions, movement speeds, etc. of the AUV and the target;
2. Read map parameters, perform environmental modeling, and set obstacles;
3. Use an external optimization algorithm to estimate the predicted encounter time t_{target} between the AUV and the t_{target} ;
4. Derive the end position, which is the predicted encounter point, from t_{target} , and perform task planning for the AUV based on this position;
5. Internal optimization aims to minimize the constraint costs and returns the corresponding actual movement time of the AUV to the intermediate optimizer;
6. The intermediate optimizer optimizes the actual movement time of the AUV to achieve the shortest time t_{auv} ;

7. Calculate the difference between the AUV movement time t_{auv} provided by the intermediate optimizer and the predicted encounter time t_{target} . Return this time difference to the external optimizer for further optimization to obtain the minimum time difference and provide the optimal predicted encounter time for both.

The flowchart of the task planning method under complete target information is illustrated in Figure 8. Under the condition of initial complete information, the parameter initialization and environment modeling are performed first. Next, the estimated encounter time is obtained through external optimization, and the end position is obtained according to the estimated encounter time. Then the constraint cost is minimized by internal optimization, and the shortest time of the actual motion is obtained by the intermediate optimizer. Finally, the difference between the two times is calculated, and the difference is returned to the external optimizer for optimization, where the optimal time for the meeting is obtained.

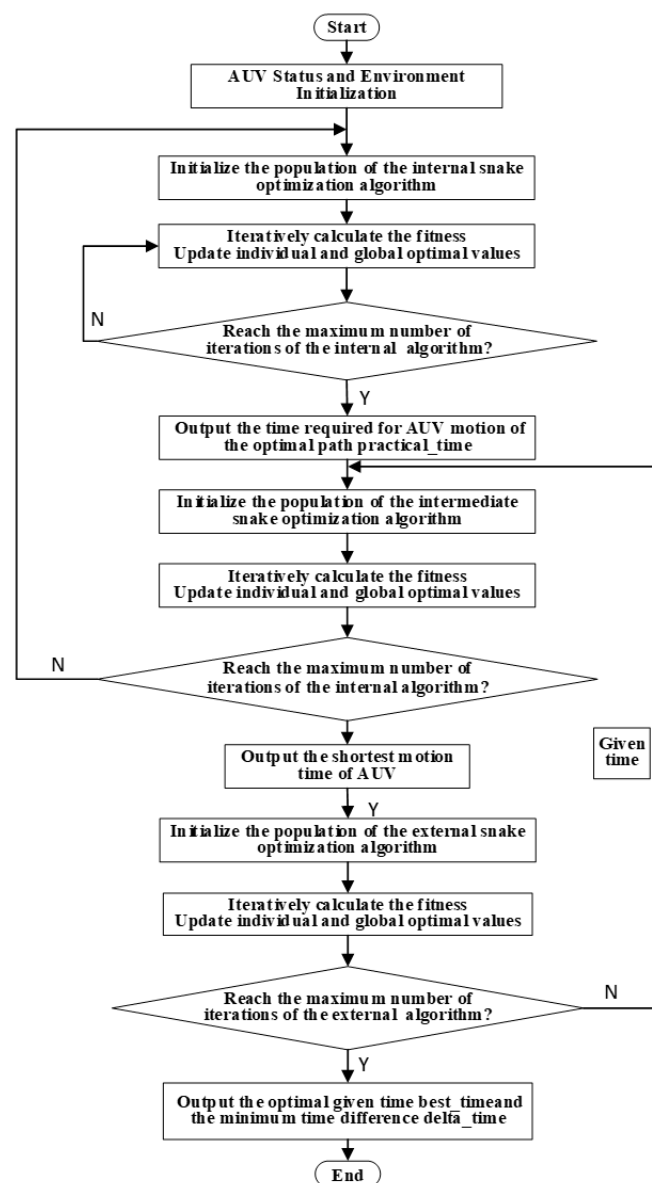


Figure 8. Complete information for the overall task planning flowchart.

The implementation of the three-level optimization task is based on the snake algorithm framework, and the main difference between the three loops lies in the objective function. The pseudocode of the path planning of the internal loop is shown in Algorithm

2. The path planning of the internal loop serves as the objective function of the external loop, achieving a three-level nested optimization. The target endpoint of the path planning of the internal loop is calculated based on the time design variables and target motion information of the external loop. The objective function of the internal loop is the fitness value calculated using Equation (19) based on the planned path. The objective function of the external loop is to calculate the error between the navigation time corresponding to the optimal path achieved through internal loop optimization under a given time optimization variable and a certain time optimization variable in the external loop. The objective function of the intermediate loop is to calculate the move time of the planning path. The pseudocode of the external and internal loop objectiveness function is in Algorithm 3.

Algorithm 2. Internal Optimization Function

```

Internal optimizer function: So_path
Input: pop_path, Iteration_path, fobj_path, dim_path, Xmin-path, Xmax-path
Output: Xfood, fval, gbest_path, path_time
1. Initialize ThresholdQ ThresholdT, c1, c2, c3
2. X = initialization(pop_path, dim_path, Xmax-path, Xmin-path)
3. Calculate fobj_path function fitness value for each Xpath
4. [GYbest, gbest_path] = min(fitness), Xfood = Xpath(gbest,:)
5. Nm = round(pop/2), Nf = pop - Nm
6. Xm = Xpath(1:Nm,:), Xf = Xpath(Nm + 1:pop,:)
7. fitness_m = fitness(1:Nm), bestXpath_m = bestX(1:Nm,:) %Male snake population
8. [fitnessBest_m, gbest1] = min(fitness_m), Xbest_m = Xm(gbest1,:)
9. fitness_f = fitness(Nm + 1:pop), bestXpath_f = bestX(Nm + 1:pop,:) %Female snake population
10. [fitnessBest_f, gbest2] = min(fitness_f), Xbest_f = Xm(gbest2,:)
11. for i = 1:Iteration
12. Temp = exp(-(t)/Iteration), Q = c1 × exp(((t - Iteration)/Iteration))
13. if Q < Threshold
14.   update Xnewm use Xm by Equation (5)
15.   update Xnewf use Xf by Equation (5)
16. else
17.   if Temp > Thresold2
18.     update Xnewm use Xbest by Equation (7)
19.     update Xnewf use Xbest by Equation (7)
20.   else
21.     if rand > 0.6
22.       update Xnewm use Xm by Equation(8)
23.       update Xnewf use Xf by Equation (8)
24.     else
25.       update Xnewm use Xm by Equation (8) replace Fm with Mm
26.       update Xnewf use Xf by Equation (8) replace Fm with Mm
27.       if rand > 0.5
28.         update Xnewm wrost by Equation (11)
29.         update Xnewf wrost by Equation (11)
30.       end
31.     end
32.   end
33. end
34. Calculate the objective function fitness value by Xnewm and update Xm
35. Calculate the objective function fitness value by Xnewf and update Xf
36. Update the optimal objective function fitness value and optimal male snake vector
37. for the male snake population
38. Update the optimal objective function fitness value and optimal female snake vector
39. of the female snake population
40. Update the global optimal objective function fitness value gbest_t
41. and global optimal snake vector Xfood
42. end

```

Algorithm 3. Objective Function of Initial Complete Target Information Conditions

Internal objective Function

Internal optimizer object function: MyCost_path

Input: $X_{\text{path}}, UUV, UUV_v$ Output: $\text{cost}, \text{path_time}$

1. $P_{\text{Terminal}} = UUV + \text{path_time} \times UUV_v$
2. $\text{Path} = [X_{\text{path}}; P_{\text{Terminal}}]$
3. Calculate path length $\text{cost } F_1$ use Path by Equation (7)
4. Calculate smoothing $\text{cost } F_2$ use Path by Equation (17)
5. Calculate threat $\text{cost } F_3$ use Path by eq(Equation)
6. Calculate fitness value cost by Equation (19)
7. Calculate path_time by Path

Intermediate objective Function

external optimizer object funtion: MyCost_time

Input: X_{time}, UUV Output: $\text{best_time}, \text{bestX}$

1. $\text{fobj_path} = @\text{MyCost_path}$
2. $[\text{bestX_path}, \text{path_time}] = \text{So_path}(\text{pop_path}, \text{Iteration_path}, \text{fobj_path}, \text{dim_path},$
3. $X_{\text{max}}, X_{\text{min}}, X_{\text{time}}, UUV)$
4. $\text{best_time} = \text{path_time}$

external objective Function

external optimizer object funtion: MyCost_deltatime

Input: X_{time}, UUV Output: $\text{delta_time}, \text{bestX}, \text{path_time}$ $\text{fobj_delta} = @\text{MyCost_delta}$
 $[\text{bestX}, \text{path_time}] = \text{So_time}(\text{pop_time}, \text{Iteration_time}, \text{fobj_time}, \text{dim_time},$
 $X_{\text{time_max}}, X_{\text{time_min}}, X_{\text{time}}, UUV)$
 $\text{delta_time} = X_{\text{time}} - \text{path_time}$ **2.4.2. Task Planning Under Conditions of Initially Incomplete Target Information**

The schematic diagram of AUV planning under the condition of initially incomplete target information is shown in Figure 9.

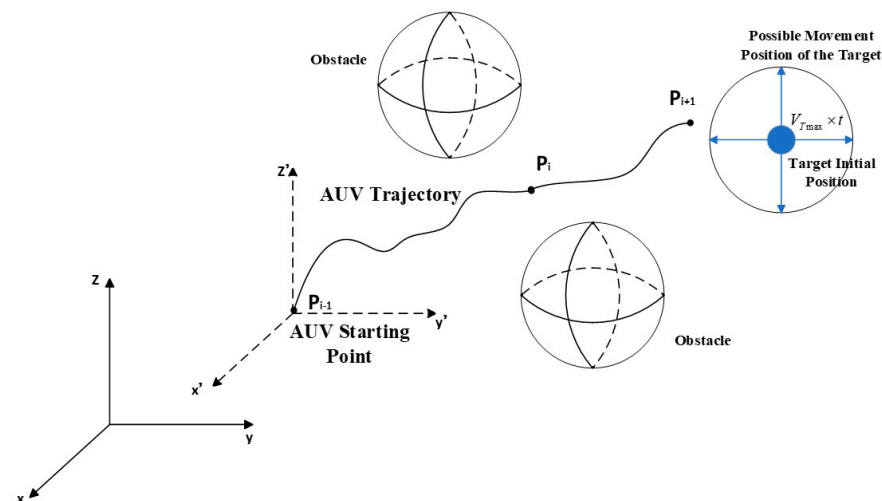


Figure 9. Schematic diagram of AUV planning under incomplete information conditions.

In this scenario, the AUV only knows the approximate initial location of the target, while the specific direction of the target's motion remains unknown. However, based on practical engineering experience, an estimation of the maximum speed of the target can be made. When the AUV is deployed from the platform, this moment is considered as time zero. At any given time t , the target's location will be somewhere within a circle centered at the initial point with a radius equal to the maximum distance the target could have traveled at its maximum speed since time zero.

The overall design approach for task planning under incomplete information is shown in Figure 10. This paper proposes a three-level optimization task planning based on the

snake optimization algorithm. Given the initial position information of the target and its maximum speed, the task planning for the AUV is carried out. In this case, the target's movement range can be approximated as a circular area. Assuming the maximum speed of the target is V_{max} , the movement time is t_1 , and the azimuth angle is θ , this information can determine the boundary of the circle. The ultimate goal is to make the AUV reach this circle in the shortest time while minimizing the size of the encircled area. The entire planning process is still divided into three levels: external planning, intermediate planning, and internal planning.

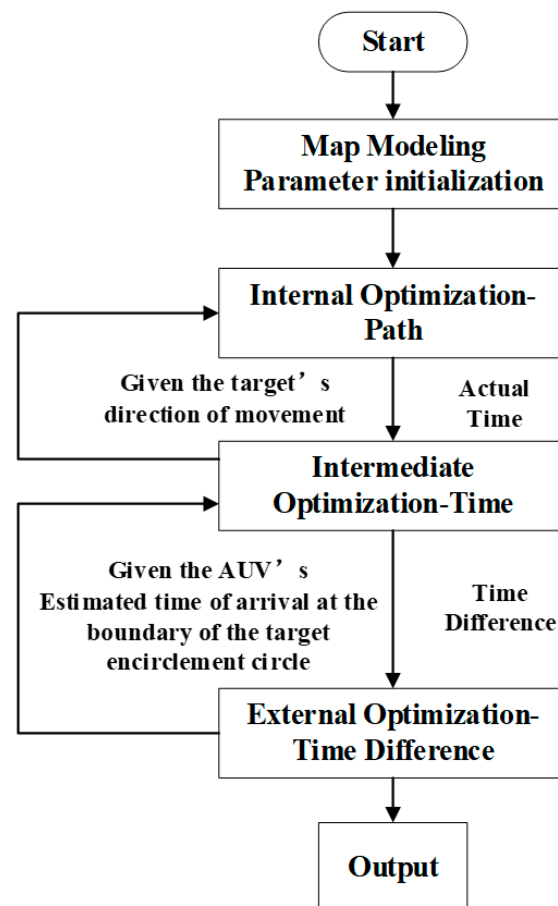


Figure 10. Overall idea diagram of three-layer task planning based on SO algorithm(incomplete target information).

External planning mainly optimizes the difference between the actual time it takes for the AUV to reach the boundary of the encircled area and the estimated time it is expected to arrive, which involves optimizing the overall objective function. Intermediate planning focuses on finding the minimum encircled area or the shortest actual movement time for the AUV given different movement azimuth angles θ of the target. Internal planning optimizes the constraints involved in AUV path planning. By continuously providing different θ values, intermediate planning aims to find the θ that results in the shortest actual movement time for the AUV or the smallest encircled area. The optimized shortest actual movement time obtained from intermediate planning is subtracted from the estimated time provided by external planning to calculate the time difference, which is then returned to external planning for further optimization. The entire process iterates until an optimal result is achieved.

Task planning under incomplete information aims to achieve the goal of the AUV reaching the smallest encircled area of the target in the shortest possible time. Therefore, the objective optimization function of external planning is taken as:

$$\Delta t = |t_{auv} - t_{give}|, \quad (20)$$

where t_{auv} represents the actual movement time of the AUV; t_{give} represents the estimated time for the AUV to reach the boundary of the encircled area; and Δt represents the difference between the actual movement time of the AUV and the estimated encounter time. The objective of this optimization function is to minimize the time difference Δt , which means making the actual movement time of the AUV as close as possible to the given estimated arrival time.

The purpose of intermediate planning is to find the direction angle at which the target moves, resulting in the smallest encircled area, which corresponds to the minimum time for the AUV to actually reach the encircled area.

Due to the unknown direction of the target's motion, the maximum dispersion circle of the target can be determined based on the given task time and maximum motion speed. If the expected angle of the AUV on the dispersion circle is determined, the endpoint of the path planning can be determined by Equations (21) and (22).

$$X_{end} = X_{ini,tar} + t_{give} * V_{max} * \sin\theta, \quad (21)$$

$$Y_{end} = Y_{ini,tar} + t_{give} * V_{max} * \cos\theta, \quad (22)$$

where $X_{ini,tar}$, $Y_{ini,tar}$ are the coordinates of the initial position of the target, V_{max} is the maximum velocity of the target, t_{give} is the expected task completion time issued by the outer loop, and θ is the design variable of the middle loop.

The internal objective optimization function is the shortest actual movement time of the AUV corresponding to the target dispersion circle angle θ . The optimal path and time to reach the endpoint of the dispersion circle can be obtained through the inner planning loop. The intermediate loop obtains the shortest navigation time and arrival angle for the AUV to meet the target dispersion circle through angle planning. The external loop optimizes the time to complete the task. The outer loop optimizes the expected task time to minimize the error between the expected task time t_{give} and the actual completion time t_{auv} .

The objective function of internal planning is the same as the internal optimization objective function under the complete information scenario.

The detailed process of the three-level optimization task planning based on the snake optimization algorithm under incomplete information conditions is as follows:

1. Initialize parameters: provide the initial positions, movement speeds, etc. of the AUV and the target.
2. Read map parameters: perform environmental modeling and set obstacles.
3. The external optimizer continuously provides the estimated encounter time (t_{give}), between the AUV and the target.
4. The intermediate optimizer continuously provides the target dispersion circle angle, θ .
5. Based on the given t_{give} and θ , task planning can be performed for the AUV.
6. Internal optimization minimizes the constraint cost and returns the corresponding actual movement time of the AUV to the intermediate optimizer.
7. By continuously adjusting the target movement azimuth angle, the intermediate optimizer finds the shortest time, t_{auv} , required for the AUV to actually move and reach the target's dispersion circle area.
8. The external optimizer calculates the difference between the actual movement time, t_{auv} , obtained from the intermediate optimization and the given time, t_{give} . Using the snake optimization algorithm, iterations are performed to continuously adjust the given time, aiming to minimize the absolute value of the difference between the actual movement time required for the AUV to reach the target's encircled area and

the given time. This ensures that the AUV can reach the boundary of the smallest encircled area of the target in the shortest possible time.

The detailed task planning process is shown in Figure 11. The mission planning process under the condition of initial incomplete target information is generally similar to that under complete target information, except that the estimated encounter time obtained by the external optimization algorithm becomes the estimated time for the AUV to reach the encirclement boundary.

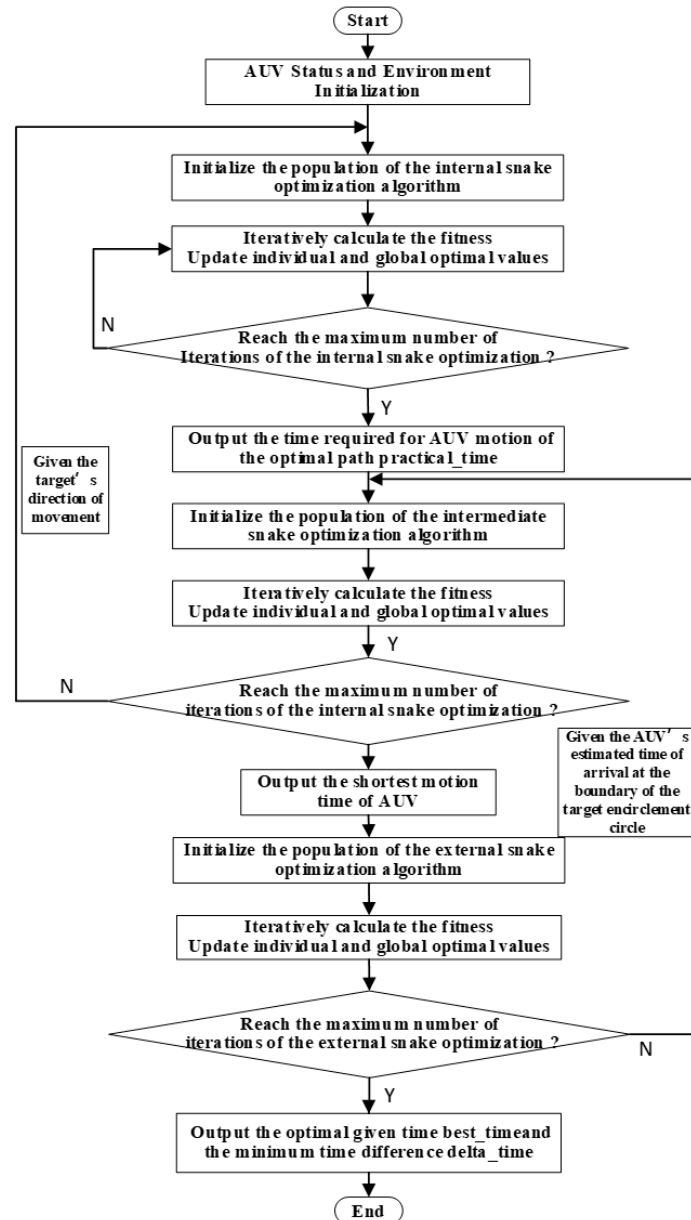


Figure 11. Incomplete information overall task planning flowchart.

The program structure of each of the three layers is basically similar (Algorithm 2). The main difference between the three layers lies in the objective function. The external layer remains the same as the complete condition, with the optimization task completion time error. The intermediate layer optimization objective function is to find the shortest navigation time path to reach the target dispersion circle and corresponding angle design variable θ , the path achieved through nested internal layer path optimization programs. The objective function of the internal layer path planning remains unchanged when compared to

the completeness condition. The pseudocode of the external and internal loop objectiveness function is in Algorithm 4.

Algorithm 4. Objective Function of Incomplete Initial Target Information Conditions

Internal objective Function

Internal optimizer object funtion: MyCost_path

Input: X_{path} , UUV , UUV_v

Output: $cost$, $path_time$

1. $P_{Terminal} = UUV + path_time \times UUV_v$
2. $Path = [X_{path}; P_{Terminal}]$
3. Calculate path length $cost F_1$ use Path by Equation (7)
4. Calculate smoothing $cost F_2$ use Path by Equation (17)
5. Calculate threat $cost F_3$ use Path by Equation (18)
6. Calculate fitness value $cost$ by Equation (19)
7. Calculate $path_time$ of Path

Intermediate objective Function

external optimizer object funtion: MyCost_theta

Input: X_{theta} , UUV , V_{max} , t_{give}

Output: $bestX$, $path_time$

1. Calculate $P_{end,tar}$ use UUV , V_{max} , t_{give} X_{theta} , by Equation (20), Equation (21)
2. $fobj_theta = @MyCost_path$
3. $[bestX, path_time] = So_path(pop_path, Iteration_path, fobj_path, dim_path,$
4. $X_{path_max}, X_{theta_min}, P_{end,tar})$

External objective Function

external optimizer object funtion: MyCost_deltatime

Input: $X_{deltatime}$, UUV

Output: $delta_deltatime$, $bestX$, $bestX_theta$, $path_time$

1. $fobj_theta = @MyCost_theta$
 2. $[bestX_theta, bestX, path_time] = So_theta(pop_theta, Iteration_theta, fobj_theta, dim_theta,$
 3. $X_{theta_max}, X_{theta_min}, X_time, X_{deltatime}, UUV)$
 4. $delta_time = X_time - path_time$
-

3. Simulation Results and Analysis

To verify the effectiveness of the proposed mission planning method, simulation experiments were conducted on an Intel i9-13900H with 32 GB of RAM (Intel, Santa Clara, CA, USA) using MATLAB R2018b. We designed simulation experiments for both complete and incomplete target information conditions separately.

3.1. Mission Planning Under Initial Complete Target Information Conditions

The input parameters for mission planning under complete information are shown in Table 3.

The initialization position and speed of AUV and target are determined by map scene size, AUV, and target motion capability. Initial Given Time can be estimated by the relative distance between the initial AUV and the target, and the algorithm will automatically iterate without affecting the final result. Path node count determines the speed and smoothness of path planning. If it is too small, it will lead to obvious path mutations, which are not conducive to obstacle avoidance and result in no solution. If it is too large, it will lead to a significant increase in planning time. Due to nested planning, it will present a geometric increase in planning time, generally ranging from 10 to 20.

The snake optimization algorithm population size and iteration count directly affect the accuracy and speed of planning; increasing them can improve the accuracy of solving the problem, facilitate escaping from local optimal solutions, reduce the speed of accelerating planning, and reduce planning time, but they will affect the accuracy of solving. When selecting, it is necessary to repeatedly try and match based on the actual situation. If the solution accuracy meets the requirements, the numerical value can be minimized to improve efficiency. If the solution accuracy does not meet the requirements or the optimal solution cannot be found, the numerical value should be increased. The range of food judgment value and temperature judgment value is $[0, 1]$, which affects the frequency of two behaviors; the typical value of food judgment value is 0.25, and the typical value of temperature judgment value is 0.6.

Table 3. Task planning input parameters under the complete information scenario.

Section	Parameter	Value
Initialization parameters	AUV initial position	[200,100,150] m
	Target initial position	[900,700,100] m
	AUV speed	10 m/s
	Target speed	0.5 m/s
	Initial given time	200 s
	Path node count	15
Internal optimization algorithm parameters	Internal snake optimization algorithm population size	50
	Internal snake optimization algorithm iteration count	100
	Food judgment value	0.25
	Temperature judgment value	0.6
	C1	0.5
	C2	0.05
	C3	2
Intermediate optimization algorithm parameters	Intermediate snake optimization algorithm population size	20
	Intermediate snake optimization algorithm iteration count	5
	Food judgment value	0.25
	Temperature judgment value	0.6
	C1	0.5
	C2	0.05
	C3	2
External optimization algorithm parameters	External snake optimization algorithm population size	20
	External snake optimization algorithm iteration count	5
	Food judgment value	0.25
	Temperature judgment value	0.6
	C1	0.5
	C2	0.05
	C3	2

Listed in the Table 3, and the detailed parameters of initialization, internal optimization algorithm, intermediate optimization algorithm, and external optimization algorithm are given. The AUV mission under complete information conditions was planned, and the software was run 10 times to obtain the simulation results shown in Table 4.

Table 4. Simulation results under complete information conditions.

Count	Best_Time (s)	Practical_Time (s)	Delat_Time (s)	Voyage (m)
1	102.3396	98.8857	3.4539	998.857
2	98.8897	98.1893	0.7004	981.893
3	102.4319	99.8661	2.5658	998.661
4	101.9350	101.3425	0.5942	1013.425
5	100.8928	100.9090	2.9838	1009.090
6	100.8224	100.7205	0.1018	1007.205
7	102.3045	101.3076	0.9969	1013.076
8	96.5964	95.4761	1.1204	954.761
9	101.6371	100.7935	0.8437	1007.935

Table 4. Cont.

Count	Best_Time (s)	Practical_Time (s)	Delat_Time (s)	Voyage (m)
10	94.8353	94.0722	0.7631	940.722
MAX	102.4319	101.3425	3.4539	1013.425
MIN	94.8353	94.0722	0.1018	940.722
MEAN	100.26847	99.15625	1.4124	992.5625

Figures 12 and 13 show the simulation diagram of AUV mission planning. The red curve represents the AUV path, and the yellow curve represents the target path. “Start” is the initial position of the AUV, “Target” is the initial position of the target, and “End” is the encounter point between the two, which is also the final destination of the target.

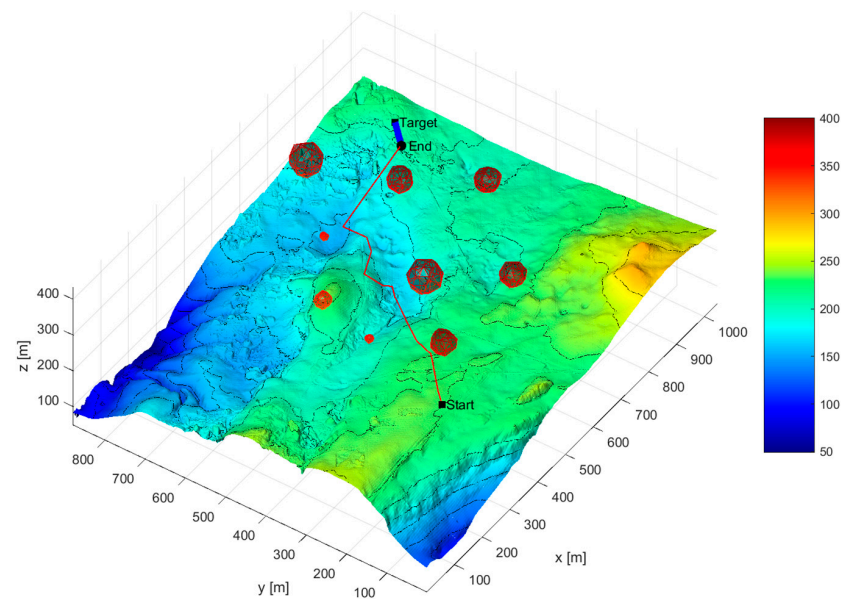


Figure 12. Complete information scenario overall task planning diagram (three-dimensional diagram).

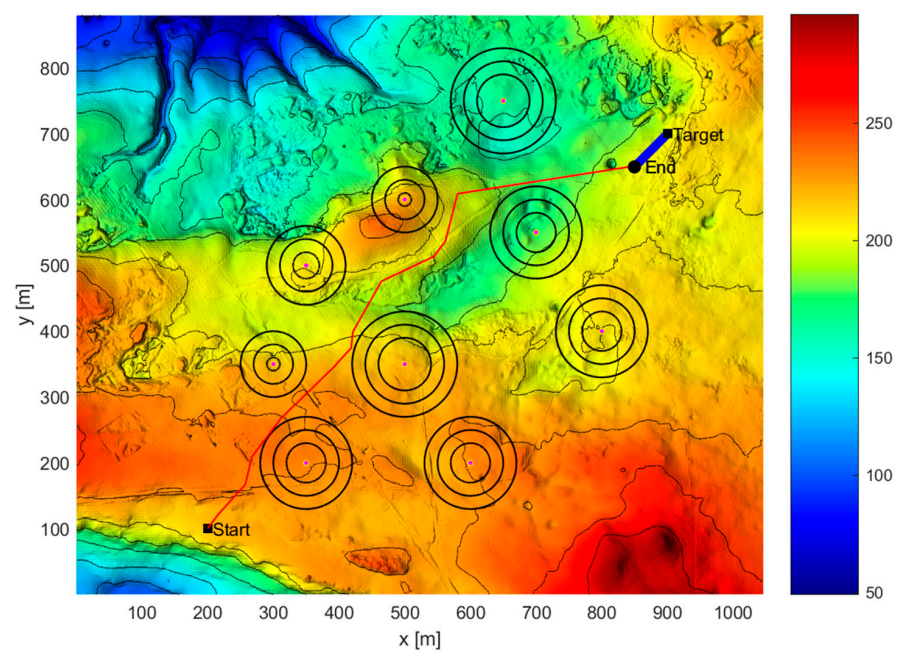


Figure 13. Complete information scenario overall task planning diagram (vertical view).

As can be seen from Table 4, under complete information conditions, the hierarchical mission planning algorithm can plan an AUV search mission path that satisfies the time threshold requirement for the deviation between the actual mission completion time and the expected mission completion time. The maximum mission planning time deviation is 3.453 s, the minimum mission planning time deviation is 0.101 s, and the average time deviation is 1.41 s. In terms of time deviation rate, the maximum time deviation rate is 3.37%, the minimum time deviation rate is 0.1%, and the average time deviation rate is 1.4%. Figures 12 and 13 show that under complete information conditions, the proposed mission planning method can plan a path that meets multiple constraints, avoids threat areas, and encounters the target as early as possible.

3.2. Mission Planning Under Initial Incomplete Target Information Conditions

The input parameters for mission planning under initial incomplete information are shown in Table 5. The input parameters of task planning under the incomplete information scenario are listed in the table, and the detailed parameters of initialization, internal optimization algorithm, intermediate optimization algorithm, and external optimization algorithm are given.

Table 5. Task planning input parameters under the incomplete information scenario.

Section	Parameter	Value
Initialization Parameters	AUV initial position	[200,100,150] m
	Target initial position	[900,700,100] m
	AUV speed	10 m/s
	target speed	0.5 m/s
	Initial given time	200 s
	Path node count	pi
Internal optimization algorithm parameters	Internal snake optimization algorithm population size	50
	Internal snake optimization algorithm iteration count	100
	Food judgment value	0.25
	Temperature judgment value	0.6
	C1	0.5
	C2	0.05
	C3	2
Intermediate optimization algorithm parameters	Intermediate snake optimization algorithm population size	20
	Intermediate snake optimization algorithm iteration count	5
	Food judgment value	0.25
	Temperature judgment value	0.6
	C1	0.5
	C2	0.05
	C3	2
External optimization algorithm parameters	External snake optimization algorithm population size	20
	External snake optimization algorithm iteration count	5
	Food judgment value	0.25
	temperature judgment value	0.6
	C1	0.5
	C2	0.05
	C3	2

Mission planning was conducted for AUVs under incomplete information conditions, and the software was run 10 times to obtain the simulation results shown in Table 6.

Table 6. Simulation results under incomplete information conditions.

Count	Best_Time (s)	Practical_Time (s)	Delat_Time (s)	Voyage (m)	Best_Theta (rad)
1	109.50	110.15	0.64	1101.46	3.69
2	112.96	111.91	1.052	1119.11	6.28
3	100.31	104.15	3.84	1041.45	4.16
4	106.98	107.53	0.55	1075.31	1.62
5	103.79	103.69	0.09	1036.94	1.33
6	99.45	99.73	0.29	997.31	2.34
7	101.17	100.78	0.38	1007.85	2.99
8	100.14	97.75	2.40	977.46	3.89
9	97.44	100.94	3.50	1009.46	4.28
10	97.84	97.04	0.8	970.37	3.62
MAX	112.96	111.91	3.84	1119.11	6.28
MIN	97.44	97.04	0.09	970.37	1.33
MEAN	103.87	104.14	1.58	1041.44	3.81

As can be seen from Table 6, under incomplete information conditions, the hierarchical mission planning algorithm can plan an AUV search mission path that satisfies the time threshold requirement for the deviation between the actual mission completion time and the expected mission completion time, while also providing the corresponding direction towards the target. The maximum mission planning time deviation is 3.84 s, the minimum mission planning time deviation is 0.09 s, and the average time deviation is 1.58 s. In terms of time deviation rate, the maximum time deviation rate is 3.6%, the minimum time deviation rate is 0.09%, and the average time deviation rate is 1.52%.

The task planning results are shown in Figures 14 and 15. The red curve is the AUV path, and the yellow curve is the target path. “Start” is the initial position of the AUV, “Target” is the initial position of the target, and “End” is the meeting point of the two, that is, the destination endpoint.

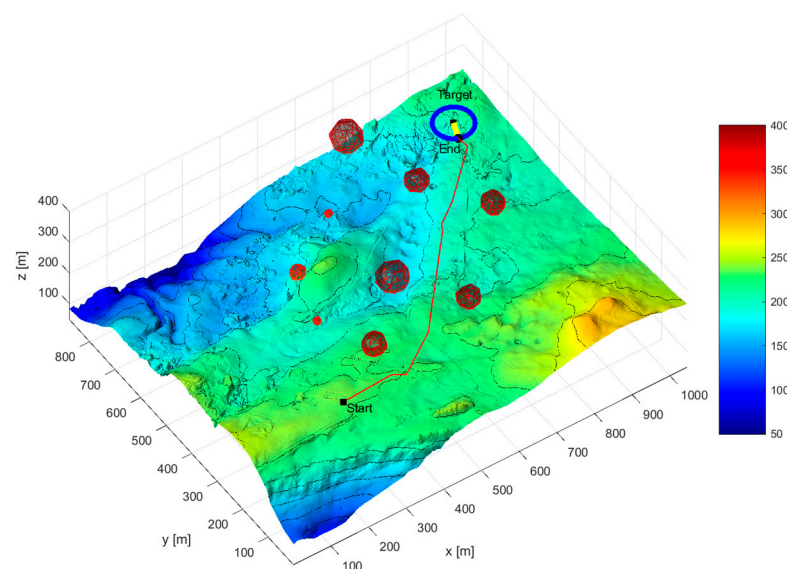


Figure 14. Incomplete information scenario overall task planning diagram (three-dimensional diagram).

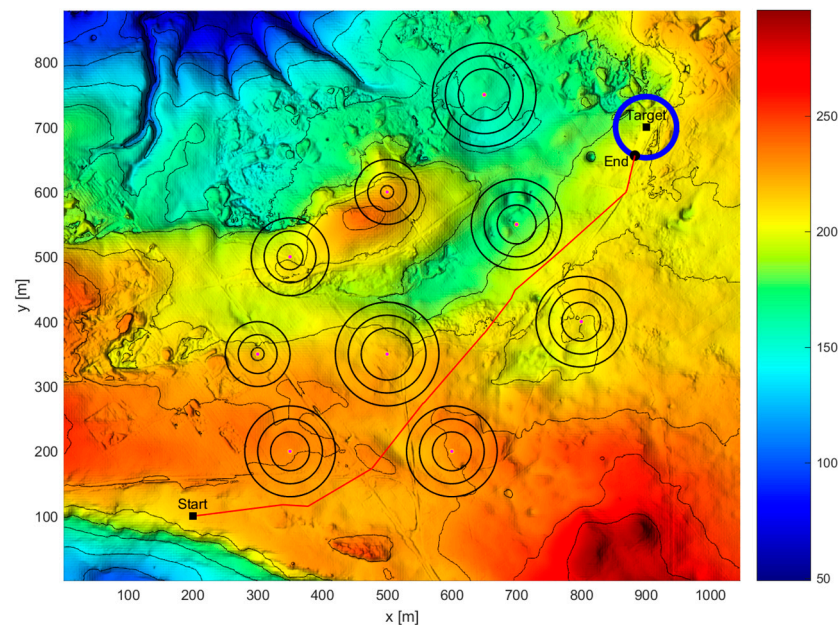


Figure 15. Incomplete information scenario overall task planning diagram (vertical view).

As shown in Figures 14 and 15, under incomplete information conditions, the proposed mission planning method can design a path that meets multiple constraints, avoids threat areas, and encounters the target as early as possible while also providing the angle at which the path enters the target dispersion area. When the AUV's movement azimuth is 3.62 radians, the resulting enclosing circle is the smallest, and the AUV can reach the target's minimum enclosing circle within 97.04 s, which is the shortest time obtained. The difference between this time and the best given time is only 0.09 s, ensuring that the AUV can encounter the dynamic target in a timely manner.

As can be seen from Table 7, due to the influence of the initial target information, the planning accuracy under complete or incomplete information conditions is not significant. In the complete information scenario, the maximum deviation rate of planning time decreases by approximately 6.9%, while in the incomplete information scenario, the average deviation rate of planning time decreases by approximately 7.8%.

Table 7. Comparison of planning performance under different information conditions.

Initial Information Conditions	Maximum Deviation Rate	Minimum Deviation Rate	Average Deviation Rate
Complete information	3.37%	0.1%	1.4%
Incomplete information	3.6%	0.09%	1.52%

4. Conclusions

In general, the task planning problem for AUVs to pursue dynamic targets can be divided into two parts: whether the AUV can encounter the dynamic target and whether the encounter time is the shortest. The mutual coupling of these two problems makes their resolution difficult. This paper proposes an AUV autonomous global task planning method by combining two typical initial observation target information scenarios. By integrating the idea of hierarchical planning with the snake optimization algorithm, the global task planning of the AUV achieves the shortest encounter time with the dynamic target. Through mathematical simulations, it is verified that the hierarchical mission planning methods proposed in this article can effectively solve the AUV global mission

planning problem, enabling the AUV to encounter dynamic targets in the shortest time possible under multiple constraints.

This study innovatively adopts a three-layer planning strategy to address the two mutually coupled problems of the AUV encountering the dynamic target and minimizing the encounter time. This enables the algorithm to obtain a global optimal solution for dynamic targets. Additionally, the novel snake optimization algorithm is introduced into the task planning of each layer, and two different planning methods are proposed based on two typical initial detection target information of the platform, allowing the algorithm to address issues related to dynamic targets and effectively solve practical engineering problems.

This study addresses the issue of underwater autonomous task planning for AUVs, providing a method for autonomous unmanned underwater vehicle task planning. Focusing on the research of methods, some simplifications are made to the environment in this study. For example, the influence of the AUV's own volume is ignored, and a fixed flight speed is set. In subsequent research, we will consider the influence of the vehicle's own factors and explore the AUV's autonomous task planning capabilities under non-uniform speed conditions.

This article proposes a hierarchical task planning method based on the snake optimization algorithm for AUV task allocation and path planning problems. Considering the coupling relationship between task allocation and path planning, the AUV task planning problem was broken down into multiple nested optimization design problems through hierarchical partitioning and solved using the snake optimization algorithm. This algorithm can effectively solve the task planning problem of AUV under different initial information conditions, providing new ideas and methods for the practical application of AUV. In future work, research should focus on improving the algorithm, enhancing its ability to adapt to complex dynamic environments, and expanding its application scenarios, such as considering the collaboration and heterogeneity of multiple drones and introducing parallel computing mechanisms to improve design efficiency.

Author Contributions: Conceptualization, Z.W. (Zhiwen Wen), Z.W. (Zhong Wang), P.W. and D.Z.; Methodology, Z.W. (Zhiwen Wen), P.W. and D.Z.; Software, Z.W. (Zhiwen Wen) and P.W.; Validation, Z.W. (Zhiwen Wen), C.N. and P.W.; Formal analysis, Z.W. (Zhong Wang), C.N. and P.W.; Investigation, C.N. and P.W.; Resources, X.W. and P.W.; Data curation, X.W. and P.W.; Writing—original draft, Z.W. (Zhiwen Wen), Z.W. (Zhong Wang), X.W., C.N. and D.Z.; Writing—review & editing, Z.W. (Zhong Wang), P.W. and D.Z.; Visualization, C.N. and P.W.; Supervision, Z.W. (Zhong Wang) and D.Z.; Project administration, Z.W. (Zhong Wang) and D.Z.; Funding acquisition, P.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Basic Scientific Research Program under Grant of JCKY2021206B005.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Written informed consent has been obtained from the patient(s) to publish this paper.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: Authors ZhiwenWen and Zhong Wang were employed by the company China State Shipbuilding Corporation Limited. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. AN/TYQ-77 Aviation Mission Planning System (AMPS) [EB/OL]. Available online: <http://www.globalsecurity.org/military/systems/aircraft/systems/antyq-77.htm> (accessed on 7 November 2012).
2. Joint Mission Planning System [EB/OL]. Available online: <http://www.globalsecurity.org/military/systems/aircraft/systems/jmps.htm> (accessed on 7 November 2012).
3. Greshler, N.; Gordon, O.; Salzman, O.; Shimkin, N. Cooperative multi-agent path finding: Beyond path planning and collision avoidance. *arXiv* **2021**, arXiv:2105.10993.
4. Zhang, Z.; Jiang, J.; Wu, J.; Zhu, X. Efficient and optimal penetration path planning for stealth unmanned aerial vehicle using minimal radar cross-section tactics and modified A-Star algorithm. *ISA Trans.* **2023**, *134*, 42–57. [\[CrossRef\]](#) [\[PubMed\]](#)
5. Zhang, R.; Guo, H.; Andriukaitis, D.; Li, Y.; Królczyk, G.; Li, Z. Intelligent path planning by an improved RRT algorithm with dual grid map. *Alex. Eng. J.* **2024**, *88*, 91–104. [\[CrossRef\]](#)
6. Guo, D.; Ji, S.; Yao, Y.; Chen, C. A decentralized path planning model based on deep reinforcement learning. *Comput. Electr. Eng.* **2024**, *117*, 109276. [\[CrossRef\]](#)
7. Wang, X.; Shi, Y.; Ding, D.; Gu, X. Double global optimum genetic algorithm-particle swarm optimization-based welding robot path planning. *Eng. Optim.* **2016**, *48*, 299–316. [\[CrossRef\]](#)
8. Wu, G.; Wang, H.; Pedrycz, W.; Li, H.; Wang, L. Satellite observation scheduling with a novel adaptive simulated annealing algorithm and a dynamic task clustering strategy. *Comput. Ind. Eng.* **2017**, *113*, 576–588. [\[CrossRef\]](#)
9. Yu, J.; Yong, E.M. Bi-level mission planning method for multi-cooperative UAV air-to-ground attack. *Syst. Eng. Electron.* **2022**, *44*, 2849–2857.
10. Roberge, V.; Tarbouchi, M.; Labonté, G. Comparison of parallel genetic algorithm and particle swarm optimization for real-time UAV path planning. *Trans. Ind. Inform.* **2012**, *9*, 132–141. [\[CrossRef\]](#)
11. Chen, X.; Reinelt, G.; Dai, G.; Spitz, A. A mixed integer linear programming model for multi-satellite scheduling. *Eur. J. Oper. Res.* **2019**, *275*, 694–707. [\[CrossRef\]](#)
12. Xiao, Y.; Zhang, S.; Yang, P.; You, M.; Huang, J. A two-stage flow-shop scheme for the multi-satellite observation and data-downlink scheduling problem considering weather uncertainties. *Reliab. Eng. Syst. Saf.* **2019**, *188*, 263–275. [\[CrossRef\]](#)
13. Ghamry, K.A.; Kamel, M.A.; Zhang, Y. Multiple UAVs in forest fire fighting mission using particle swarm optimization. In Proceedings of the International Conference on Unmanned Aircraft Systems, Miami, FL, USA, 13–16 June 2017; pp. 1404–1409.
14. Wang, B.; Ye, D.; Sun, Z.W.; Tang, S.; Chen, X. Hierarchical planning for on-orbit self-reconfiguration of modular reconfigurable satellites. *Acta Aeronaut. Astronaut. Sin.* **2019**, *40*, 322912.
15. Bai, Y.; Kanellakis, C.; Nikolakopoulos, G. SA-reCBS: Multi-robot task assignment with integrated reactive path generation. *IFAC-PapersOnLine* **2023**, *56*, 7032–7037. [\[CrossRef\]](#)
16. Han, S.D.; Yu, J. Optimizing space utilization for more effective multi-robot path planning. In Proceedings of the International Conference on Robotics and Automation (ICRA), Philadelphia, PA, USA, 23–27 May 2022; pp. 10709–10715.
17. Chen, Z.H.; Li, J.; Liu, C.; Li, J.; Liu, X.Y. Task planning method for coordinated attacks on ground targets under time constraints. *Syst. Eng. Electron.* **2023**, *45*, 2353–2360.
18. Chen, Y.; Yang, D.; Yu, J. Multi-UAV task assignment with parameter and time-sensitive uncertainties using modified two-part wolf pack search algorithm. *IEEE Trans. Aerosp. Electron. Syst.* **2018**, *54*, 2853–2872. [\[CrossRef\]](#)
19. Wang, Z.; Liu, L.; Long, T.; Wen, Y. Multi-UAV reconnaissance task allocation for heterogeneous targets using an opposition based genetic algorithm with double-chromosome encoding. *Chin. J. Aeronaut.* **2018**, *31*, 339–350. [\[CrossRef\]](#)
20. Jia, Z.; Yu, J.; Ai, X.; Xu, X.; Yang, D. Cooperative multiple task assignment problem with stochastic velocities and time windows for heterogeneous unmanned aerial vehicles using a genetic algorithm. *Aerosp. Sci. Technol.* **2018**, *76*, 112–125. [\[CrossRef\]](#)
21. Liu, J.; Anavatti, S.; Garratt, M.; Abbass, H.A. A hierarchical mission planning system for multi-uncrewed ground vehicles using fast cost evaluation and ant colony optimization. *Inf. Sci.* **2024**, *679*, 121029. [\[CrossRef\]](#)
22. Bi, W.; Zhang, M.; Chen, H.; Zhang, A. Cooperative task allocation method for air-sea heterogeneous unmanned system with an application to ocean environment information monitoring. *Ocean Eng.* **2024**, *309*, 118496. [\[CrossRef\]](#)
23. Zhu, D.; Cao, X.; Sun, B.; Luo, C. Biologically inspired self-organizing map applied to task assignment and path planning of an AUV system. *IEEE Trans. Cogn. Dev. Syst.* **2018**, *10*, 304–313. [\[CrossRef\]](#)
24. Zhang, R.; Li, S.; Ding, Y.; Qin, X.; Xia, Q. UAV path planning algorithm based on improved Harris Hawks optimization. *Sensors* **2022**, *22*, 5232. [\[CrossRef\]](#)
25. Yu, J.; Chen, Z.; Zhao, Z.; Yao, P.; Xu, J. A traversal multi-target path planning method for multi-unmanned surface vessels in space-varying ocean current. *Ocean Eng.* **2023**, *278*, 114423. [\[CrossRef\]](#)
26. Han, G.; Lai, W.; Wang, H.; Zhu, S. Hybrid-Algorithm-Based Full Coverage Search Approach with Multiple AUVs to Unknown Environments in Internet of Underwater Things. *IEEE Internet Things J.* **2024**, *11*, 11058–11072. [\[CrossRef\]](#)
27. Zou, W.; Zou, J.; Sang, H.; Meng, L.; Pan, Q. An effective population-based iterated greedy algorithm for solving the multi-AGV scheduling problem with unloading safety detection. *Inf. Sci.* **2024**, *657*, 119949. [\[CrossRef\]](#)
28. Han, X.; Cheng, W.; Meng, L.; Zhang, B.; Gao, K.; Zhang, C.; Duan, P. A dual population collaborative genetic algorithm for solving flexible job shop scheduling problem with AGV. *Swarm Evol. Comput.* **2024**, *86*, 101538. [\[CrossRef\]](#)

29. Cui, Y.; Jia, B.; Sang, H.; Meng, L.; Zhang, B.; Zou, W. An Effective Discrete Jaya Algorithm for Multi-AGVs Scheduling Problem with Dynamic Unloading Time. *IEEE Access* **2024**, *12*, 101701–101716. [[CrossRef](#)]
30. Hashim, F.A.; Hussien, A.G. Snake Optimizer: A novel meta-heuristic optimization algorithm. *Knowl.-Based Syst.* **2022**, *242*, 108320. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.