Problem Chosen	2024 MCM/ICM Summary Sheet	Team Control Number
B		**2409202**

Searching for Submersibles: Comprehensive and Efficient Submarine Rescue System for MCMS

Summary

Maritime Cruises Mini-Submarines (MCMS) aims to lead tourists on underwater adventures in the Ionian Sea using mini-submarines. Safety is a paramount concern for the company. The objective of this paper is to develop a comprehensive submarine rescue model. This model ensures that, in the event of communication interruption, mechanical failure, or loss of power, the hosting ship and rescue ship can efficiently search and rescue the mini-submarine. We propose a submarine positioning model and a search and rescue model, validating their effectiveness through data analysis, visualization modeling, and other techniques.

Initially, we obtain bathymetric data of the Ionian Sea within a specific rectangular range (36N20E to 37N21E) with a resolution of 15° using GEBCO 2030. Based on this seabed data, we construct a 3D model.

To predict the location of a missing submarine, we employ the Markov Chain Monte Carlo (MCMC) method to simulate the possible positions of the missing submarine. Results indicate the effectiveness of this method in determining the potential locations under the influence of multiple factors.

In the search and rescue model, the predicted location of the missing submarine serves as the center of the search range, expanding spirally outward until the submarine is located.

During the search, the hosting ship releases sonar to explore potential submarine locations, often resulting in numerous false positives. To minimize rescue time, we propose the use of unmanned submersibles to search all samples in the shortest time.

The unmanned submersible search problem is treated as the Traveling Salesman Problem (TSP). We optimize TSP using the Genetic Algorithm (GA) to obtain the shortest path.

Additionally, we establish dynamic models for the rescue ship and hosting ship during the search process, calculating the spiral path on the sea surface to maximize search efficiency. Finally, we develop a kinematic model for the rescue submarine to assess rescue success rates under different predictive scenarios.

Keywords: Ionian Sea; submarine, hosting ship; rescue submarine; MCMC; TSP; GA; kinematic model

Contents

1 Introduction

1.1 Background

Maritime Cruises Mini-Submarines (MCMS) is a Greek company dedicated to providing mini-submarine services. MCMS offers its customers a unique and exhilarating underwater exploration experience with their Mini-Submarines. The submarines are towed from the hosting vessel to the destination and deployed using wireless tethers. MCMS is now looking to utilize their submarines to lead tourists on underwater expeditions in the Ionian Sea, searching for shipwreck remains. In order to ensure the safety of the visitors and obtain regulatory approval, it is crucial to develop safety procedures to address potential communication disruptions with the hosting vessel and potential mechanical failures, including loss of submarine power.

1.2 Restatement of the problem

Our task is to develop a comprehensive submarine rescue model for MCMS company to address communication disruptions between the submarine and the hosting vessel, as well as potential mechanical failures, including loss of submarine power. The objective is to quickly locate the faulty submarine in order to ensure the safety of the tourists.

Therefore, this problem can be analyzed in four parts:

1) Develop a model that can forecast the submarine's position in time, considering the uncertainties involved in the prediction, to facilitate the localization of the faulty submarine.

2) Select additional equipment to be installed on both the hosting vessel and rescue vessel, taking into account the cost of salvage operations and other relevant factors.

3) Develop a model that utilizes the information from the position prediction model to recommend the initial deployment points and search patterns for the equipment, aiming to minimize the time required to locate the lost submarine. Determine the probability of finding the submarine based on time and cumulative search results.

4) Evaluate and discuss the developed model and address issues related to its implementation and promotion.

1.3 Literary review

Significant advancements have been made in submarine positioning, navigation, and communication technologies, providing higher levels of accuracy, reliability, and efficiency in submarine operations.

In terms of positioning, the Global Positioning System (GPS) has become a primary means of accurately locating submarines by transmitting position information through satellite signals. Additionally, Inertial Navigation Systems (INS) combined with map and sensor information enable precise navigation of submarines [1].

In the realm of communication, acoustic communication technology is widely applied to achieve long-distance and bidirectional communication. Sonar systems are used to transmit sound and receive echoes [2], serving purposes such as target detection, communication, and navigation. Furthermore, the development of underwater fiber-optic communication [3] and underwater acoustic radio communication technology [4] has provided new options for submarine communication.

The continuous improvement and innovation of these technologies will further enhance the positioning, navigation, and communication capabilities of submarines in underwater environments.

1.4 Our Work

Our work primarily focuses on the following aspects:

1) Selecting the Indian Ocean from the GEBCO 2023 dataset and obtaining high-resolution underwater topographical data within the rectangular range of 36N20E to 37N21E to serve as the actual background for the problems in this paper.

2) Implementing 3D modeling based on data-driven approaches and utilizing the Markov Chain Monte Carlo method to simulate the possible location of the lost submarine. We analyze the uncertain factors that affect positioning and propose rational recommendations.

3) Considering cost factors, we employ cost-benefit analysis and equipment selection criteria as methodologies. Based on the actual situation, we determine the additional search equipment to be carried by the host ship and the rescue equipment required to be carried by the rescue ship.

4) We optimize the search path for the rescue submersible using a genetic algorithm, significantly reducing rescue time. We propose two spiral search paths for the main ship and the rescue ship on the sea surface based on the predicted location of the submarine. Furthermore, we establish a dynamic model of the search and rescue process based on the actual location of the submarine, which can be used to assess the success rate of rescue under different prediction accuracies.

To avoid complex descriptions and visually represent our workflow, the flowchart is shown in Figure 1 below.

2 Notations and Assumptions

2.1 Notations

Table 1: Notations used in this paper

Symbol	Description
Pc	Crossover Probability in the Genetic Algorithm
Pm	Mutation Probability in the Genetic Algorithm
Path_ori	Initial Shortest Search Path
Path_op	Shortest Search Path after GA Optimization
Ii	Individual i in the Genetic Algorithm Population
$\alpha(x' \mid x)$	Probability of transitioning from current state x to new state x'
$q(x' \mid x)$	Proposed probability of transitioning from state x to x'
f	Coriolis parameter
V	Geostrophic current
$\cdot V$	Flow velocity

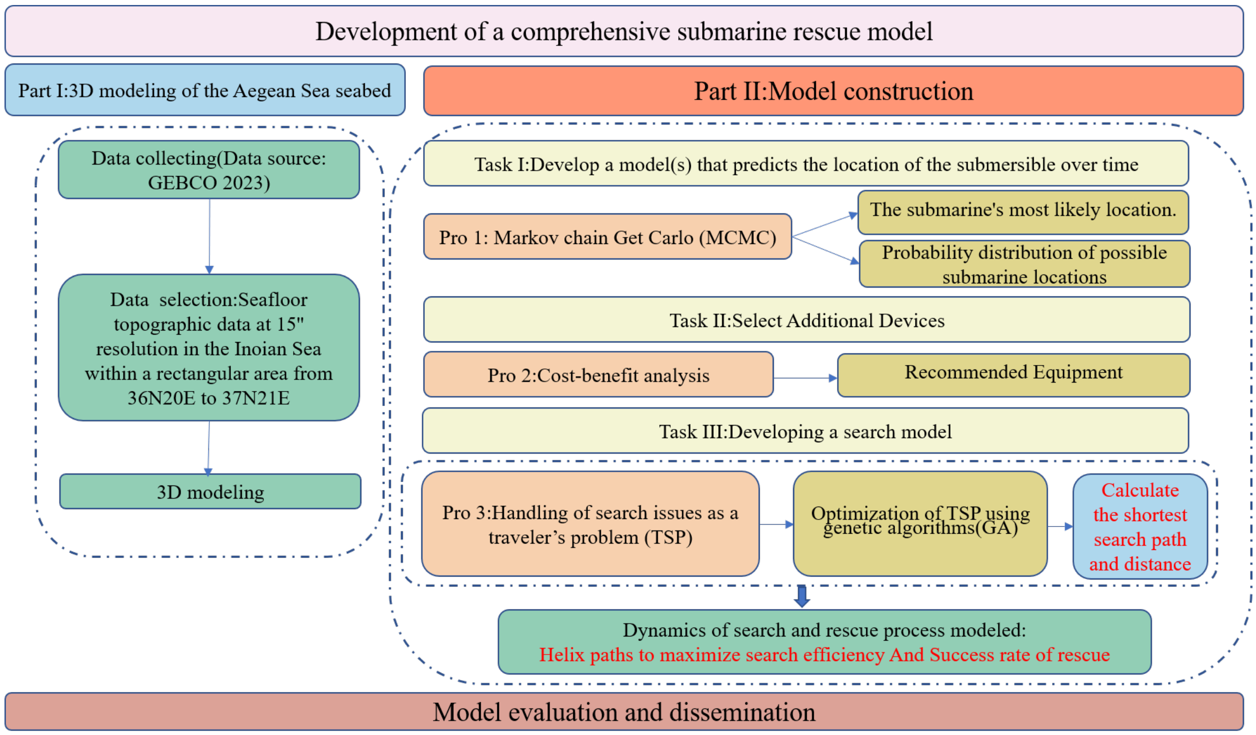

Figure 1: Workflow chart

2.2 Assumptions

1. Assumption of environmental stability: We assume that the underwater environment within the specific rectangular range (36N20E to 37N21E) of the selected Ionian Sea remains relatively stable over a short period of time, allowing us to use fixed environmental parameters in the model.

2. Assumption of data accuracy: We assume that the data transmitted by the submarine, such as equipment status and depth, is accurate or that the range of error is known. The underwater data obtained in this paper and the references are assumed to be correct.

3. Assumption of operator competence: We assume that the operators on board the malfunctioning submarine, the host ship, and the rescue ship all perform their operations correctly.

4. Assumption of equipment functionality: We assume that the detection equipment on board the host ship and the rescue ship, such as the active sonar detector and unmanned submersibles, are functioning properly.

3 Model 1: positioning submersible model

3.1 Analysis of the Problem

For Problem 1, we need to develop a model that predicts the position of a submarine over time and incorporate devices to reduce the uncertainty associated with the predictions.

3.2 Model Introduction

According to our analysis, the current position of the submarine is subject to random influences stemming from various factors. To model the transition from one state to another, we have decided to utilize the Markov Chain Monte Carlo (MCMC) method. To account for the uncertainty in these state transitions, we will incorporate the impact of ocean currents on the submarine's behavior.

3.2.1 MCMC

The Markov Chain Monte Carlo (MCMC) method is a random sampling technique [5] that enables efficient sampling from complex probability distributions, facilitating probabilistic inference and parameter estimation for addressing problems that traditional methods struggle to handle. Consequently, MCMC has become a crucial tool in fields such as statistics, machine learning, and Bayesian analysis.

MCMC comprises two key components: the Markov Chain and the Monte Carlo method. The Markov Chain is a stochastic process where the future state depends solely on the current state and is independent of past states. As the chain iterates for a sufficient length of time, the distribution of states stabilizes, ceasing any significant changes and reaching a stationary distribution. Monte Carlo involves estimating the target variable by randomly generating sample points and leveraging their statistical properties to approximate the numerical solution to the problem.

MCMC integrates the Markov process with Monte Carlo simulation, allowing for dynamic simulations where the sampling distribution evolves as the simulation progresses, overcoming the limitation of traditional Monte Carlo methods, which are only capable of static simulations. The defining factor of this method lies in the transition matrix (or transition probability matrix) that governs the probabilities of transitioning between different states in the Markov chain.

Common MCMC algorithms, such as the Metropolis-Hastings algorithm, employ a mechanism to propose new states and determine whether to accept these states based on acceptance probability rules. The algorithm involves generating a potential new state, calculating the acceptance probability, and then accepting the new state with a certain probability or maintaining the current state. Through iteratively updating states, this process eventually achieves sampling from the target distribution. The probability formulas involved are as follows:

Proposal Probability:$q\left(x' \mid x\right)$represents the probability of transitioning from the current state x to a new state x'.

Acceptance Probability:

$$\alpha\left(\mathrm{x}' \mid \mathrm{x}\right) = \min\left(1, \frac{p\left(x'\right) q\left(x \mid x'\right)}{p(x) q\left(x' \mid x\right)}\right) \tag{1}$$

In this context, $p(x)$ refers to the probability density of the target distribution at state x, and $q\left(x' \mid x\right)$ represents the proposal probability of transitioning from state x to x'.

3.2.2 Uncertainty

The underwater vehicle in the deep sea is subject to various unpredictable influences, such as geostrophic currents, which result from the Earth's rotation and give rise to the Coriolis effect.

This effect causes deviations in the water flow, potentially causing the trajectory of the underwater vehicle to be less straight than expected.

To account for the impact of geostrophic currents on the underwater vehicle, we make adjustments to its state behavior. The foundational equation is as follows:

$$\mathrm{V} = \frac{-g}{f} \times \frac{\Delta h}{\Delta x} \tag{2}$$

Here, V represents the flow velocity, g denotes the acceleration due to gravity, f represents the Coriolis parameter (which is latitude-dependent, as explained below), Δh represents the height difference at the sea surface (often proportional to the underwater pressure difference), and Δx denotes the horizontal distance.

The Coriolis parameter, denoted as f, is a parameter associated with the Earth's rotation that describes the influence of the Coriolis force on the motion of fluids, such as the atmosphere and the oceans. The Coriolis parameter is directly related to the latitude of a specific point on Earth. Its expression is given as:

$$f = 2\Omega \sin \varphi \tag{3}$$

Here:Ω represents the Earth's angular rotation rate, which is approximately 7.2921e 10-5 radians per second,φ denotes the latitude, where higher latitudes correspond to larger values of the Coriolis parameter. Positive values are assigned to latitudes in the northern hemisphere, while negative values are assigned to latitudes in the southern hemisphere.

3.3 Modeling

Using the aforementioned model, we retrieve bathymetric data with a resolution of 15 within the rectangular region of 36N20E to 37N21E in the Inoian Sea. We then generate a topographic map as the background for the entire problem. We initialize the position and motion state data of the underwater vehicle before its loss of communication, taking into account the normalcy of its propulsion system.

Subsequently, we employ Markov Chain Monte Carlo (MCMC) sampling for the underwater vehicle's position at each time step. This sampling approach ensures equal likelihood for each position, discarding the samples that do not satisfy the prescribed conditions and only accepting those that meet the criteria. This process allows us to obtain all possible outcomes. The algorithm, as described below, outlines these steps:

1) Retrieve bathymetric data within the specified region (36N20E to 37N21E) with a resolution of 15.

2) Generate a topographic map based on the retrieved data.

3) Initialize the underwater vehicle's position and motion state data accounting for normal propulsion.

4) Perform MCMC sampling for each time step of the underwater vehicle's position.

5) Discard the samples that do not satisfy the prescribed conditions.

6) Accept only the samples that meet the criteria, resulting in all possible outcomes.

Algorithm 1: Pseudocode of Position Prediction

Set simulation parameters, Initialize submersible position(X_s0,Y_s0,H_s0), movement time $t0$ movement speed $v0$, Lost time $num\_motions$;
Critical code: Perform MCMC sampling on the position of the submersible at each moment;
for *each num_motion m* **do**
    $initialize\_current\_motion(m)$;
    for *each time step i (from 2 to N+1)* **do**
        $propose\_new\_position()$;
        $calculate\_acceptance\_probability()$;
        if *random number <= acceptance_probability* **then**
            $accept\_new\_position()$;
        if *random number > acceptance_probability* **then**
            $reject\_new\_position()$;

3.4 Model results

Under different propulsion scenarios, a model for predicting the underwater vehicle's position over time was established. In the event of the vehicle's loss of propulsion, potential position predictions are depicted in Figure2. Each point in the figure represents a simulated result, indicating the predicted position of the underwater vehicle at different time points. The colors of these points are used to differentiate between various prediction outcomes or represent different time stages. Additionally, a surface is employed to express relevant geographical information, providing a more intuitive understanding of the underwater vehicle's motion trends in the geographical space. By combining the scatter plot and the surface representation, a comprehensive and visual analysis and interpretation of the position prediction results of the underwater vehicle can be achieved, extracting information regarding the vehicle's behavior and environmental influences.

From the figure, it can be observed that the data points are densely distributed, reflecting potential movement paths of the underwater vehicle or possible descent trajectories following the loss of propulsion. This provides crucial information about the behavior of the vehicle, indicating the positional relationships simulated at different time points.

When the underwater vehicle retains propulsion, potential position predictions are displayed in Figure3, illustrating the results of positional inferences under the scenario of propulsion failure. Compared to the situation where there is no propulsion, the distribution of scatter points is more dispersed, and the transition of behavioral states becomes more random. This indicates that when propulsion is lost, accurately predicting the position of the underwater vehicle becomes more challenging, resulting in a wider search area and potentially longer rescue times. Therefore, it is crucial for the underwater vehicle to regularly transmit information to the host ship before any incidents occur. This information includes uncertain factors that influence the position of the underwater vehicle in this model, such as the velocity and direction of seawater currents, differences in height and horizontal distance from the sea surface, as well as the current motion data and underwater topography of the vehicle. This enables more accurate positioning of the

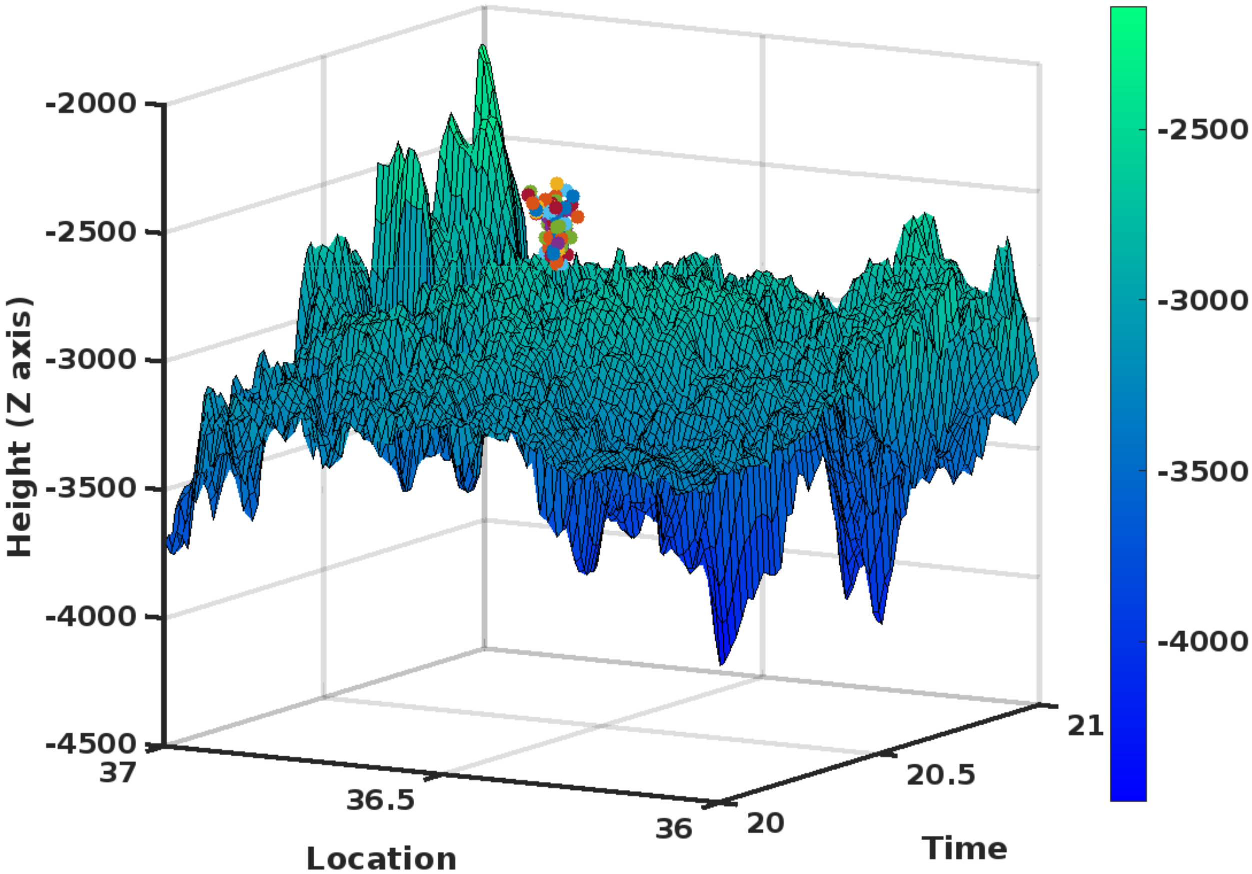

Figure 2: Prediction of Possible Submersible Positions after Power Loss

underwater vehicle.

4 The need to carry additional facilities

4.1 Analysis of the Problem

Determining the additional search equipment to be carried by the hosting vessel and the rescue equipment needed to be carried by the rescue vessel, considering cost factors.

4.2 Methodology of Problem Solving

4.2.1 Methodology Description

1) Cost-Benefit Analysis: It is a method to evaluate decision options by comparing their economic benefits. In the context of search equipment, cost-benefit analysis considers various costs associated with purchasing, maintenance, preparation, and utilization of these devices, and compares them with the devices' search efficiency.

2) Equipment Selection Criteria: Based on a set of key indicators such as coverage range, detection depth, accuracy, and reliability of the equipment, combined with the results of cost-benefit analysis, a comprehensive equipment selection strategy can be developed.

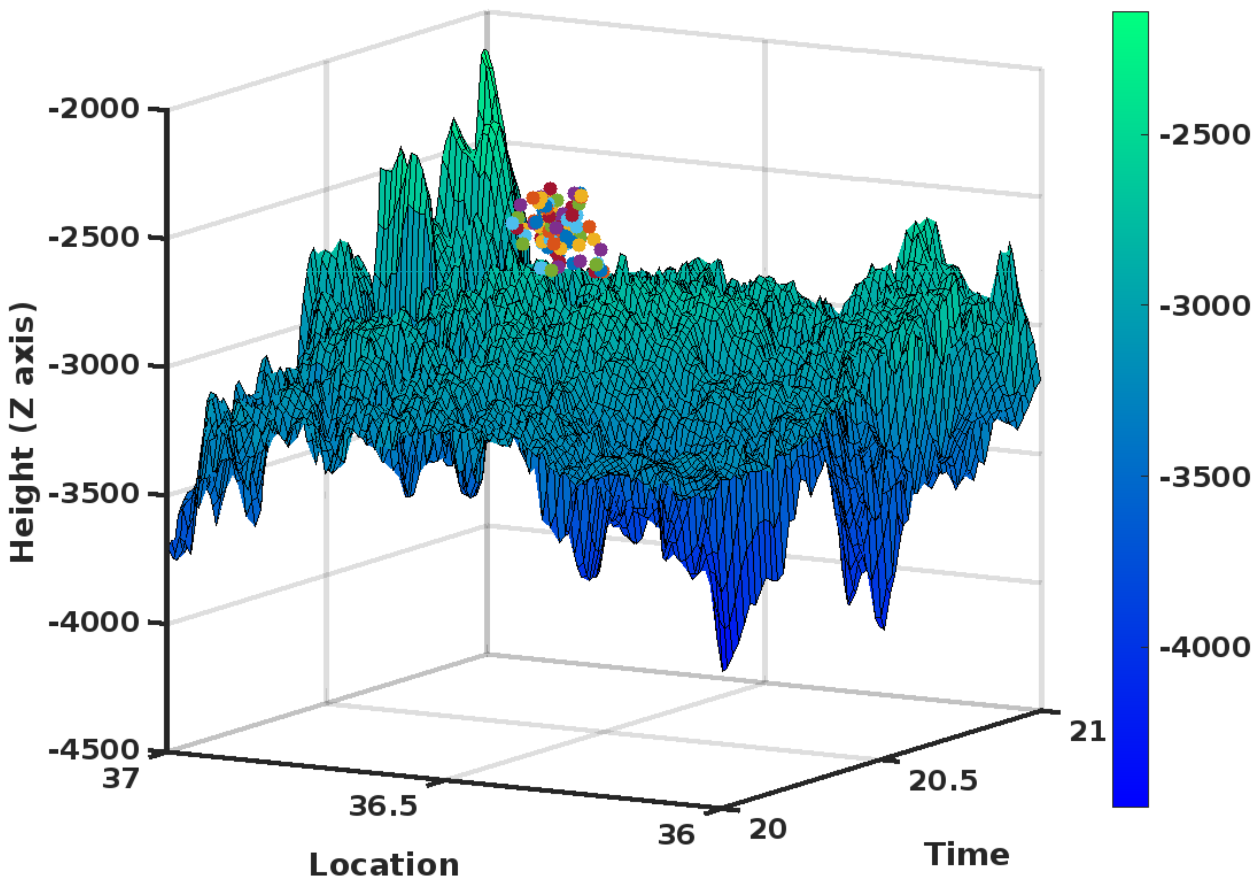

Figure 3: Prediction of Possible Submersible Positions after Power Restoration

4.2.2 Mathematical Mode

Based on the methodology description mentioned above, the utility function$U(e)$ for the devices can be defined as a function of both search efficiency and cost:

$$U(e) \quad = \quad \alpha^{*}E(e) - \beta * C(e) \tag{4}$$

Here, e represents different search devices, $E(e)$ denotes the search efficiency of the device (e.g., coverage range, depth, etc.), $C(e)$ represents the total cost of the device (including purchasing, maintenance, and usage costs), α and β are weighting parameters used to adjust the relative importance of search efficiency and cost in the utility function. The total cost $C(e)$ can be further refined as:

$$C(e) \quad = \quad C \text{ purchase } (e) + C \text{ maintenance } (e) + C \text{ readiness } (e) + C \text{ usage } (e) \tag{5}$$

which correspond to the costs associated with device purchase, maintenance, readiness state, and usage.

4.3 Device Recommendation

In order to improve the search and rescue operations, we recommend the following devices:

1) Unmanned Underwater Vehicle (UUV): An UUV should be installed on the cargo ship. This device will assist in locating the malfunctioning underwater robot by conducting targeted searches and providing additional data for accurate positioning.

2) Active Sonar Search Devices: A set of active sonar search devices should also be installed on both the cargo ship and the rescue vessel. These devices utilize sound waves to detect and locate objects underwater, significantly enhancing the search capabilities in uncertain situations.

By incorporating these recommended devices, the search and rescue efforts will be more effective, increasing the chances of successful outcomes.

5 Recognition and Retrieval of Submersibles

5.1 Search Equipment

Taking into account factors such as time and cost associated with retrieving submersibles, the model prescribes the tugboat to be equipped with a rescue submersible and a set of sonar search devices. The rescue ship, on the other hand, should carry four rescue submersibles and two sets of active sonar search devices to enhance search and rescue efficiency.

Upon losing communication with the submersible, it is imperative to initiate search and rescue operations. The entire rescue process can be simplified into three steps. Firstly, activate the active sonar search system to conduct an extensive search in the areas where the submersible is most likely to be, capturing the sonar signals emitted by the submersible. Secondly, using small rescue submersibles, confirm the sonar signals of suspected submersibles and determine their actual locations. Finally, employ the rescue submersibles' towing ropes to secure the incident submersible and safely hoist it using winches on the rescue ship.

5.2 Search and Rescue Model — Rescue Submersibles

5.2.1 Problem Analysis

Planning the search path based on sonar detection of potential submersible locations constitutes a typical Traveling Salesman Problem (TSP). In this study, it involves searching various possible locations of the submersible, ensuring an optimal search path. Common methods to solve TSP include exhaustive search, greedy algorithms, dynamic programming, simulated annealing, genetic algorithms, etc. The advantages and disadvantages of these methods are summarized in Table 1.

Table 2: Algorithm Characteristics

Algorithm	Advantages	Disadvantages	Characteristics
Exhaustive Search	Global optimum	Computation cost rises with problem size	Small-scale problems
Greedy Algorithm	High efficiency	No global optimum guarantee	Specific scenarios
Dynamic Programming	Avoids redundant calculations	High modeling complexity	Problems with overlapping subproblems
Simulated Annealing	Global search capability	Parameter sensitivity	Complex search spaces
Genetic Algorithm	Global search capability	Algorithm parameter sensitivity	Large-scale problems, complex search spaces

Genetic Algorithm (GA) is an optimization algorithm inspired by natural selection and genetic mechanisms, widely applied to search and optimization problems. Its core idea is to simulate the process of natural evolution, generating new individuals through genetic operations within a population, gradually optimizing solutions within the solution space.

In this study, the search and rescue submersible problem is inherently a complex optimization problem involving a vast search space and uncertainty. Therefore, applying Genetic Algorithm to solve this problem exhibits several advantages, such as:

•Global Search and Local Optimization: Genetic Algorithm can maintain multiple solutions to prevent getting stuck in specific local optima.

•Adaptability and Evolution: Genetic Algorithm can adapt better to the complexity of search and rescue tasks through evolutionary processes.

•Parameter Tuning: Genetic Algorithm parameters (such as crossover rate, mutation rate, etc.) are relatively easy to adjust, allowing flexible tuning based on the nature and requirements of the problem, thereby enhancing the algorithm's robustness.

•Diversity and Parallelism: Genetic Algorithm maintains diverse individuals in the search space, contributing to improved global search performance.

5.2.2 Model Introduction

The Genetic Algorithm seeks the optimal solution to a problem by simulating the process of biological evolution. In this study, the utilization of Genetic Algorithm to obtain the optimal path for searching submersibles involves the following steps, as illustrated in Algorithm 1.

• Initialization of Population: If the number of locations to be searched is denoted as D, the path for searching all locations can be represented as a D-dimensional vector I, with I serving as an individual. Randomly generate a set of individuals to form the initial population P.

$P = \{I_1, I_2, \ldots I_N\}$;

Where I_i is the i-th individual, and N is the population size.

• Fitness Evaluation: Calculate the fitness of each individual in the population.

• Selection: Based on the fitness values of individuals, employ a roulette wheel selection method to choose two parents. Individuals with better fitness have a higher probability of being selected, simulating the process of natural selection.

• Crossover: Apply crossover operations to the two parents, generating new individuals, as shown in Algorithm 2.

• Mutation: Apply mutation operations to the produced new individuals to increase the diversity of the population, as shown in Algorithm 3.

• Replacement: Compare the new individuals with the original ones. If the fitness is superior, replace the original individuals.

• Iteration: Repeat the above steps iteratively, outputting the ultimately found individual, representing the optimized solution to the problem.

5.2.3 Results Analysis

We implemented the optimization of the TSP problem based on the Genetic Algorithm (GA) using MATLAB. The parameter settings were as follows: a population size of 1500, an individual

Algorithm 2: Pseudocode of Search Path Evolution

Set the probability of crossover and mutation P_c and P_m, generation g, tournament size num_Tour, optimal search path op_path

Initialization: Population p's individuals by generating search paths through randomization;

for *each generation* g **do**

  $all\_sum\_path \leftarrow$ fitness of all individuals;

  $op\_path \leftarrow \min(all\_sum\_path)$;

  for *each individual* **do**

    $parent1, parent2 \leftarrow$ tournament(*all individuals*, all_sum_path, num_Tour);

    $offspring \leftarrow$ crossover($parent1, parent2, P_c$);

    $offspring \leftarrow$ mutate($offspring, P_m$);

    if *fitness*($offspring$) > *fitness*($individual$) **then**

      $individual \leftarrow offspring$;

Algorithm 3: Pseudocode of Crossover

Input: *parent1, parent2* (selection by tournament strategy);

Output: *offspring*;

if *random* < *Pc* **then**

  $w = parent1.size()/2$;

  $pos = random(w)$;

  $offspring(pos : pos + w) = parent1(pos : pos + w)$;

  $offspring(other) = parent2(other)$;

else

  $offspring = parent_1$;

size equivalent to the number of locations to be searched set at 20 (deemed to align with realistic sonar detection results), a crossover probability of 0.8, and a mutation probability of 0.2. Figure 4(a) illustrates the convergence curve during the iteration process.

From the graph, it is evident that the GA method reaches convergence around 15 iterations, showcasing rapid convergence and the ability to find the optimal solution. The optimized shortest search path decreased by 69.58% compared to the initial result, as depicted in Figure 4(a). This substantial reduction in search path length is poised to significantly diminish the rescue time for the submersible, affirming the effectiveness of the GA method in optimizing the search and rescue process.

Optimization Parameters: In our experiments, we observed significant effects of different population sizes on optimization results. Therefore, we examined the impact of various population sizes on optimization results, as depicted in Figure 4(b). The optimal proportion is defined as follows:

optimal proportion = $(Path_{ori} - Path_{op}) / Path_{ori}$

Algorithm 4: Pseudocode of Mutate

Input: *offspring*;
Output: *offspring*;
for L *in offspring* **do**
  if *random* $< P_m$ **then**
    $L_{swap} \leftarrow$ randi(offspring);
    swap(L, L_{swap});

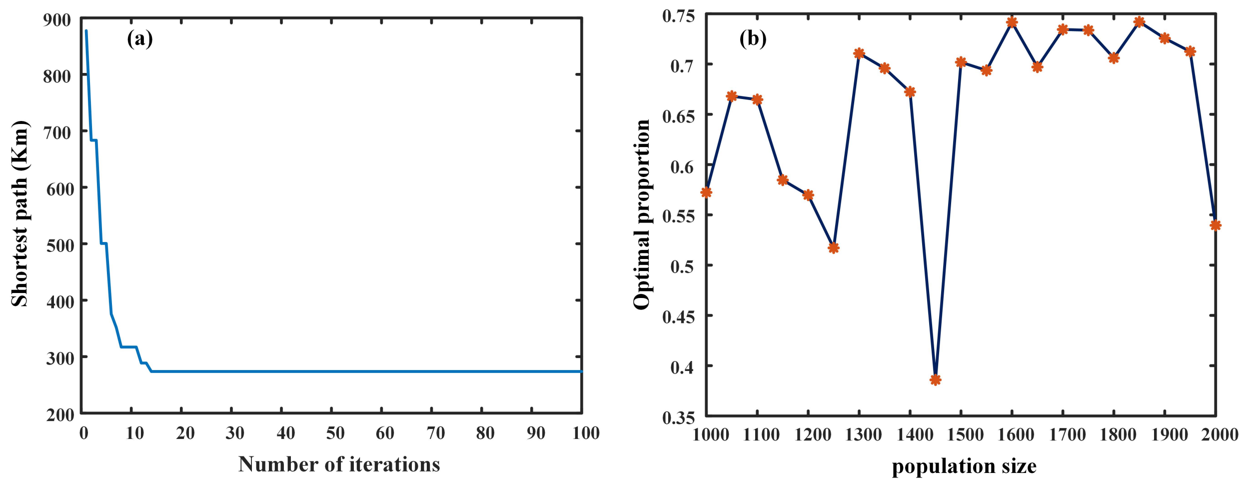

Figure 4: GA-Optimized TSP (a) Changes in the Shortest Path during the Iterative Process (b) Impact of Different Population Sizes on Optimization Results

Where $Path_{ori}$ is the initial shortest search path, and $Path_{op}$ is the shortest search path after GA optimization. Figure4(b) shows the population size ranging from 1000 to 2000 (determined based on experimental experience). From the corresponding optimization results, it is evident that population sizes with better optimization effects are mostly concentrated between 1600 and 1900. Consequently, we ultimately chose 1750 as the population size parameter.

To visually represent the effectiveness of the GA algorithm in solving the TSP problem in this study, we utilized the aforementioned parameter settings to obtain optimization results, presented in a three-dimensional format in Figure 5.

Clearly visible in Figures 5(a) and 5(b) are the initial and optimized search paths, respectively. It is evident that after optimization with the GA algorithm, the search path becomes more concise and clear, reducing additional search costs. This enhancement is instrumental in quickly locating the submersible.

5.3 Search and Rescue Model — Tugboat

In the event of a submersible incident, the tugboat will promptly initiate rescue operations, targeting the nearest location within the search area. Simultaneously, the rescue ship will depart from the shore to reach the incident area. Initially, the search will focus on the maritime areas where

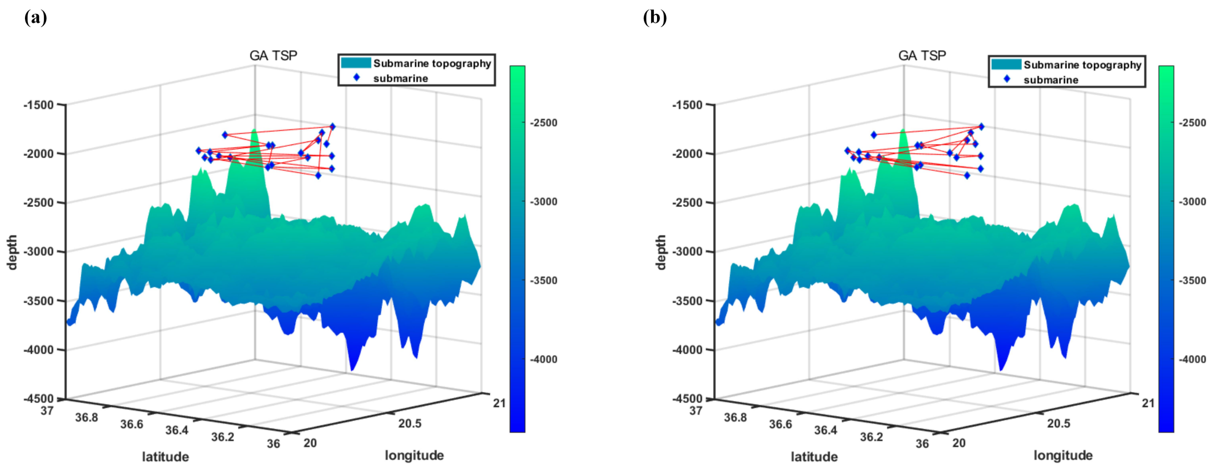

Figure 5: Visualization of GA-Optimized TSP Results (a) Initial Search Path (b) Optimized Search Path

the submersible is most likely to appear, as predicted by the model proposed in this study. The search will be conducted by rescue submersibles, utilizing the shortest search path in conjunction with the active sonar search system. This approach covers the entire seabed area within the predicted region. If the submersible is located in this area, a successful and rapid rescue can be achieved.

If the active search system fails to locate the submersible in the estimated area, the search scope needs to be expanded. The tugboat or the rescue ship will adopt a spiral movement pattern centered around the predicted region, continually expanding the search outward until the submersible is found or the search time limit is reached. When expanding the search area, the movement routes of the tugboat and rescue ship are illustrated in Figure 6(a) when the submersible is stationary. When the predicted position of the submersible changes, the movement routes of the tugboat and rescue ship dynamically adjust according to the predicted route of the submersible, as shown in Figure 6(b).

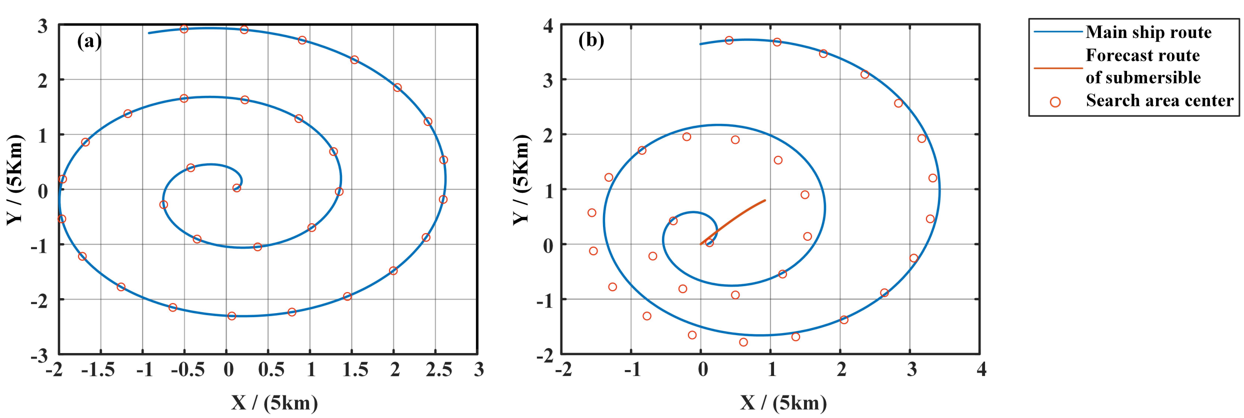

Figure 6: Search Routes of Tugboat or Rescue Ship (a) Search Routes when the Submersible is Stationary (b) Search Routes when the Submersible is in Motion

5.4 Dynamics Model of Submersible Retrieval

After the rescue submersible locates the submersible, it needs to be lifted to the sea surface. For this purpose, we have developed a dynamic model for the capture of the rescue submersible. The model includes the movement of the main ship/rescue ship, active sonar search, and the dynamics of both the rescue submersible and the distressed submersible. Figure 5 illustrates two scenarios during the submersible search. Figure 7 represents the search and rescue model when the submersible is located in the predicted area. From the figure, it can be observed that the submersible can be successfully lifted to the sea surface by the rescue submersible.

Figure 8 illustrates the scenario when the submersible is not in the predicted area. In this case, the sonar cannot detect the submersible, and the main ship/rescue ship needs to conduct a spiral search.

In this model, the active sonar continuously adjusts its position to cover the search area, aiding in the localization of the submersible. The unmanned submersible performs different actions in two scenarios: it floats when stationary and adjusts its depth while moving to simulate search tasks. The entire process is controlled by a series of parameters, including ocean current speed, submersible horizontal speed, initial positions, and velocities of the rescue ship and submersible. Through visualization, we can observe the trajectories of each element under different conditions, providing a dynamic showcase and analysis for the rescue mission.

This simulated rescue process not only demonstrates the collaboration among participants but also takes into account various factors such as seafloor topography, sonar search, and submersible status. It serves as a valuable reference and visualization tool for real-life rescue missions.

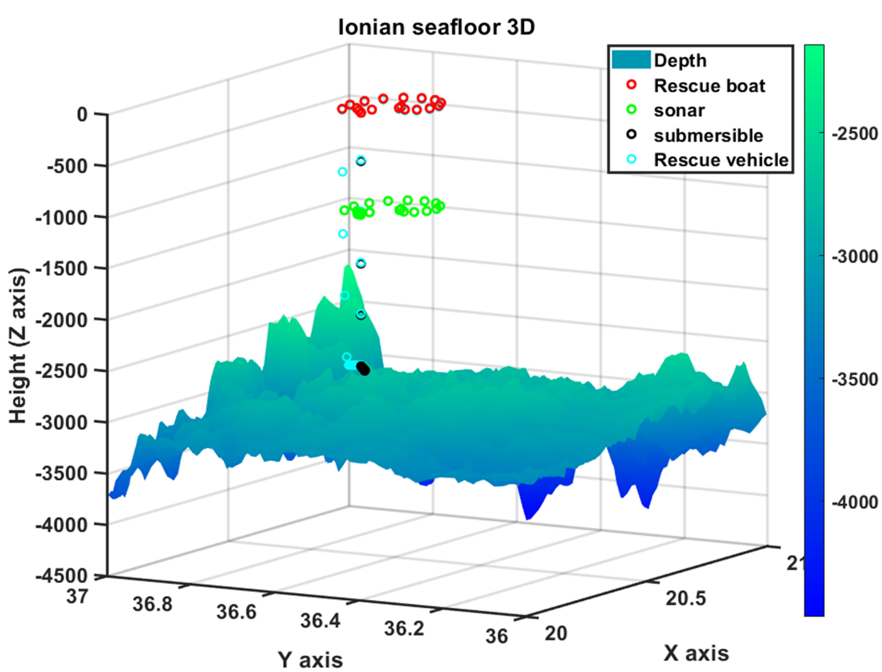

Figure 7: Capture Dynamics Model of the Rescue Submersible : submersible in the Predicted Area

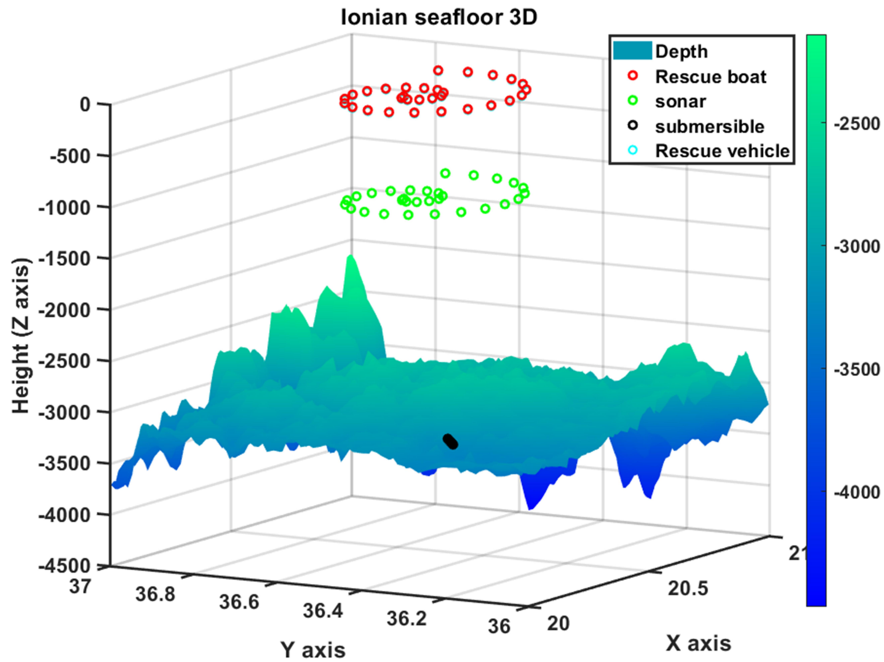

Figure 8: Capture Dynamics Model of the Rescue Submersible : submersible not in the Predicted Area

6 Model evaluation

Advantages: 1) Efficiency: By utilizing the Markov Chain Monte Carlo method, the model simulates the probability distribution of the missing submarine's location and identifies the most likely position to be found. This helps rescue personnel to efficiently locate the submarine, reducing search time and resource consumption.

2) Depth information: The 3D modeling of the seafloor using GEBCO 2030 data provides a more realistic understanding of the seafloor terrain. This enables more accurate position predictions, thereby enhancing the success rate of rescue operations.

3) Adaptability of multiple models: The submarine rescue model incorporates a genetic algorithm (GA) to optimize the Traveling Salesman Problem (TSP), allowing the model to adapt to different search conditions and constraints. It quickly identifies false positional information in the results of active sonar searching by minimizing the travel distance.

4) Dynamics model: By establishing a dynamics model for the optimal search path of rescue and host ships on the sea surface, the model provides strategies for vessel movements in the rescue process, maximizing search and rescue efficiency. **Disadvantages:** 1) Data resolution limitations: Despite utilizing the 3D modeling with data provided by GEBCO 2030, which has a resolution of 15°, it may not provide sufficiently accurate seafloor terrain information. This limitation could potentially impact the accuracy of submarine location prediction and the effectiveness of the rescue process.

2) Prediction accuracy: Although the model calculates the success rate of the rescue by simulat-

ing the motion states of the rescue and accident submarines, it still relies on accurate predictions of the submarines and the underwater environment. If the predictions are inaccurate, the effectiveness and reliability of the model may be affected.

7 Extrapolated outlook of the model

1) Application to other tourist destinations: If applied to the Caribbean Sea, for example, it would be necessary to obtain data on ocean current speeds, underwater pressure, and seafloor terrain of that region as the accuracy of the model predictions highly depends on the input data. Additionally, factors affecting the underwater operations, such as different oceanic flows compared to the Aegean Sea, should be taken into account, which may impact the submarine localization and the search pattern for rescue ships. Therefore, adjustments to the model parameters would be needed to adapt to the new environment.

2) Multi-submarine operations: For multi-submarine operations, considerations should be given to coordinate search and rescue missions of multiple submarines to effectively cover the search area and improve search efficiency. A collaborative search model would enable submarines to share information and optimize the overall search strategy. Thus, the introduction of multi-objective tracking algorithms (e.g., genetic algorithms for multi-objective optimization) would be required to handle the prediction and tracking of multiple submarines in the same area.

Figure9 depicts a simplified motion of two submarines within the Caribbean Sea region, illustrating their respective trajectories. In practical applications, the movement of multiple submarines becomes considerably complex, incorporating influences from water currents, interactive forces, and attempts to maintain a safe distance to avoid collisions.

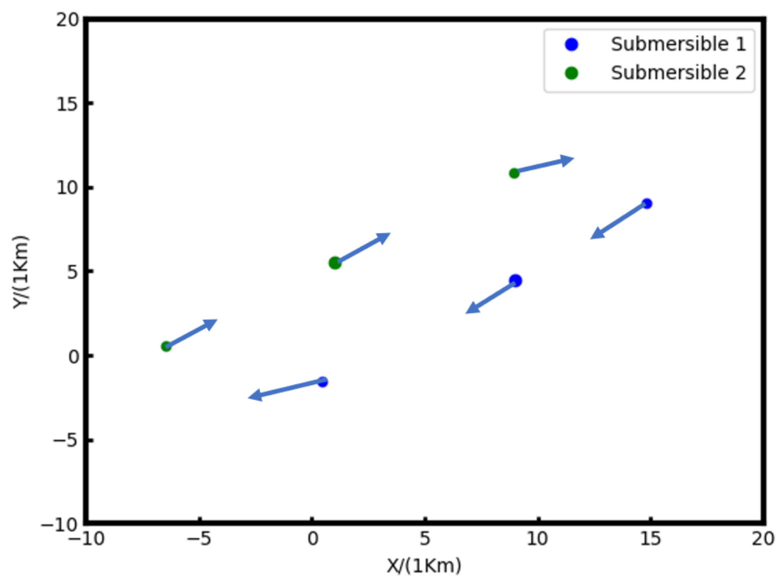

Figure 9: Simulation of the simplified movement of two submersibles in the Caribbean Sea area

8 Summarize

The model constructed in this paper provides a comprehensive solution for locating, identifying, and rescuing submarines in case of a communication failure. It combines terrain data analysis, MCMC prediction, GA algorithm optimization, and dynamic modeling to offer a deep understanding and innovative approach to submarine rescue in complex marine environments. This model accurately predicts the submarine's location and optimizes rescue strategies to enhance the efficiency of locating and rescuing submarines. By integrating various data and technologies, it equips rescue personnel with a comprehensive tool to better respond to emergency situations and maximize the survival chances of submarine crew members. In summary, this model offers a comprehensive approach to address submarine communication failure and rescue, providing crucial decision support for stakeholders.

Our model primarily relies on terrain data within a specific region in the Ionian Sea (36N20E to 37N21E) and conducts three-dimensional modeling. By employing Markov Chain Monte Carlo (MCMC) methods, we simulate the possible positions of the lost submarine, obtaining a probability distribution of its potential locations and identifying the most probable positions. During rescue operations, the mother ship deploys active sonar to search for potential submarine locations, followed by close-range identification using unmanned submersibles. We treat this problem as a Traveling Salesman Problem (TSP) and optimize it using Genetic Algorithms (GA). Simultaneously, we establish dynamic models for the search process and provide optimal search paths for the rescue and mother ships on the water surface. Furthermore, by simulating the motion states of the rescue submarine and the incident submarine, we demonstrate the success rate of rescue under varying prediction accuracies.

The model developed in this research utilizes various methods and technologies to provide a comprehensive and innovative solution for addressing the challenges of submarine rescue in complex marine environments. The approaches and principles of this model can be applied to other marine regions and various types of submarine rescue missions. Future work could further explore the application of the model in multi-submarine operations, safety and risk assessment, as well as regulatory compliance, thereby examining its potential use in new diving destinations. By integrating research in these areas, we can offer a more comprehensive, sustainable, and safe submarine rescue solution that provides enhanced support for rescue operations and diving activities.

9 Memo

To: The Greek Government Offcials.
From:Team # 2409202.
Subject: Comprehensive and Efficient Submarine Rescue System for MCMS
Date: February 5, 2024.

Dear Government Officials:

On behalf of Maritime Cruises Mini-Submarines (MCMS), I am pleased to submit a memorandum regarding our developed submarine rescue model. Our company aims to utilize our manufactured submarines for underwater exploration and wreckage discovery in the Ionian Sea. Ensuring the safety of tourists and obtaining regulatory approval are our foremost priorities. Consequently, we have urgently developed a comprehensive submarine rescue model to address potential communication disruptions between the submarines and the mother ship, as well as mechanical failures, including the loss of power in the submarines.

Our model primarily relies on specialized terrain data and three-dimensional modeling specific to the designated area in the Ionian Sea (36N20E 37N21E). We employ the Markov chain Monte Carlo (MCMC) method to simulate the potential location of a missing submarine, allowing us to generate a probability distribution of likely locations and identify the most probable location for the submarine. By analyzing the probability distribution and the most likely location, we can determine the region to which the rescue submarine needs to navigate. This results in improved search and salvage efficiency, reducing both time and resource waste. Additionally, by deploying rescue vessels and equipment based on the simulated location probability distribution, we ensure precision in our deployment strategy. Submarine salvage operations are complex and hazardous, and our model's accurate deployment plan maximizes the safety of rescue personnel.

The submarine rescue process entails three main stages. Firstly, the mother ship deploys active sonar technology to search for potential locations of the missing submarine. These potential locations are then closely inspected by unmanned underwater vehicles (UUVs) to address the issue of false location information. To facilitate this inspection, we treat the problem as a traveling salesman problem (TSP) and optimize it using genetic algorithms (GAs). Subsequently, the accident submarine is secured using the towing ropes carried by the UUVs, and it is lifted using the winch on the rescue vessel. To maximize search and rescue efficiency, we have developed a dynamic model for the search process and provided the optimal search paths for both the rescue vessel and the mother ship on the sea surface. Furthermore, by simulating the motion states of the rescue and accident submarines, we can demonstrate the rescue success rates under different prediction accuracy levels.

After discussing the advantages and improvements of our model, we further address topics such as model extrapolation, multi-submarine operations, safety and risk assessment, as well.

Sincerely yours,

Team # 2409202

References

[1] X. Sun, Z. Yang, J. Wang, and J. Yuan. A new underwater navigation solution based on integration of gps and ins. *Sensors*, 2019.

[2] H. Gao, S. Wei, and L. Xu. Underwater target localization using forward scattering. *IEEE Transactions on Signal Processing*, 2018.

[3] A. Durak and M. S. Alouini. Underwater optical wireless communications: Advances and prospects. *IEEE Communications Surveys & Tutorials*, 2020.

[4] C. Song, Y. Hong, and X. Xing. Navigating netted underwater miniature auvs using cooperative lbl and acoustic channel sharing. *IEEE Transactions on Industrial Electronics*, 2020.

[5] Stephen Brooks. Markov chain monte carlo method and its application. *Journal of the royal statistical society: series D (the Statistician)*, 1998.

Appendices

Here are simulation programmes we used in our model as follow.

Input matlab source:

```
%%Positioning submersible model
Dth = xlsread('.\Ionian.xlsx');
Dp = Dth(:, 3:5);
dy = 0.0042;
dx = 0.0042;
x = Dp(:, 2);
y = Dp(:, 3);
z = Dp(:, 1);
[X, Y] = meshgrid(linspace(min(x), max(x), 100), ...
                  linspace(min(y), max(y), 100));
Z = griddata(x, y, z, X, Y, 'cubic');
figure;
surf(X, Y, Z); hold on;

% Title and axis labels
title('Ionian seafloor 3D');
xlabel('X axis');
ylabel('Y axis');
% xlim([20.25 20.75])
% ylim([36.25 36.75])
zlabel('Height (Z axis)');
L = 111e3; % One degree distance
Xs0 = 20.3944 * L * cos(36.5251/180 * pi); Ys0 = 36.5251 * L; Hs0 = -2500;
vt0 = -pi/4; vyl0 = 20000; vylt0 = 0; vh0 = -25;
% Submersible with propulsion
vs0 = 300000;
% Submersible without propulsion
% vs0 = 30;
num_motions = 100; % Loss of contact time
N = 1000; % Number of samples for each motion
sigma = 100; % Standard deviation of random perturbation
positions = zeros(num_motions, N, 3); % Store positions for all motions
v = zeros(num_motions, N, 5); % Store velocity information for all motions

% MCMC sampling for the position of the submersible at each time
```

```
step
for m = 1:num_motions
    % Initialize the starting position for the current motion
    positions(m, 1, 1) = Xs0;
    positions(m, 1, 2) = Ys0;
    positions(m, 1, 3) = Hs0;
    v(m, 1, 1) = vs0;
    v(m, 1, 2) = vt0;
    v(m, 1, 3) = vh0;
    v(m, 1, 4) = vyl0;
    v(m, 1, 5) = vylt0;

    for i = 2:N
        % Propose a new position
        t1 = sigma * randn / 10;
        v1 = sigma * randn;
        t2 = sigma * randn / 10;
        v2 = sigma * randn / 25;
        v3 = 2 * sigma * randn;
        proposed_position(1) = positions(m, i-1, 1) + min(max(v(m, i, 1) + v3, 0), vs0) * cos(v(m, i, 2) + t1) + (v(m, i, 4) + v1) * cos(v(m, i, 5) + t2);
        proposed_position(2) = positions(m, i-1, 2) + min(max(v(m, i, 1) + v3, 0), vs0) * sin(v(m, i, 2) + t1) + (v(m, i, 4) + v1) * sin(v(m, i, 5) + t2);
        proposed_position(3) = positions(m, i-1, 3) + (v(m, i, 3) + v2);

        % Calculate acceptance probability (modify based on actual scenario)
        alpha = min(1, probability_density(proposed_position) / probability_density(positions(m, i-1)));

        % Accept or reject
        if rand <= alpha
            positions(m, i, 1) = proposed_position(1);
            positions(m, i, 2) = proposed_position(2);
            positions(m, i, 3) = proposed_position(3);
            v(m, i, 1) = min(max(v(m, i, 1) + v3, 0), vs0);
            v(m, i, 2) = v(m, i, 2) + t1;
            v(m, i, 2) = min(max(v(m, i, 2) + v2, -100), 100);
            v(m, i, 4) = min(max(v(m, i, 4) + v1, 0), vyl0);
            v(m, i, 5) = v(m, i, 5) + t2;
        else
            positions(m, i, :) = positions(m, i-1, :);
```

```
                v(m, i, :) = v(m, i-1, :);
            end
        end
end

% Plot trajectories for all motions
for m = 1:10:num_motions
    for k = 1:100:N
        for i = 1:size(Dp, 1)
            cnt = 0;
            if Dp(i, 2) - positions(m, k, 1) / L / cos
               ↪ (36.5251/180 * pi) < dx && Dp(i, 3) - positions
               ↪ (m, k, 2) / L < dy
                positions(m, k, 3) = max(positions(m, k, 3), Dp(i
                   ↪ , 1) + 10);
                cnt = cnt + 1;
            end
            if cnt > 3
                break;
            end
        end
        scatter3(positions(m, k, 1) / L / cos(36.5251/180 * pi),
           ↪ positions(m, k, 2) / L, positions(m, k, 3)); hold
           ↪ on;
    end
end
hold off;
title('Possible Positions of the Submersible');
xlabel('Time');
ylabel('Position');

function p = probability_density(x)
    % Normal probability density function
    p = exp(-0.5 * x.^2); % Normal distribution
end
```

```
%%Model 2 Part Code
%%%%%%%%%%%% Custom Parameters %%%%%%%%%%%%
tStart = tic; % Algorithm timer
[~, cities] = Read('dsj1000.tsp');
cities = cities';
cityNum = 10;
cities = cities(:, 1:cityNum);
maxx = max(cities(1, :));
minx = min(cities(1, :));
```

```
maxy = max(cities(2, :));
miny = min(cities(2, :));
for i = 1:cityNum
    cities(1, i) = (cities(1, i) - minx) / (maxx - minx) * 0.5 + 20.25;
    cities(2, i) = (cities(2, i) - miny) / (maxy - miny) * 0.5 + 36.25;
end
dep = 100 * randn(1, cityNum) - 2000;
cities(3, :) = dep;
% cityNum = 100;
maxGEN = 1000;
popSize = 100; % Population size for genetic algorithm
crossoverProbability = 0.9; % Crossover probability
mutationProbability = 0.1; % Mutation probability
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

gbest = Inf;
% Get all potential submersible positions
% cities = rand(2, cityNum) * 100; % 100 is the maximum distance

% Calculate distances for the generated cities
distances = calculateDistance(cities);

% Generate population, where each individual represents a path
pop = zeros(popSize, cityNum);
for i = 1:popSize
    pop(i, :) = randperm(cityNum);
end
offspring = zeros(popSize, cityNum);
% Save the minimum paths for each generation to facilitate plotting
minPaths = zeros(maxGEN, 1);

% Genetic Algorithm
for gen = 1:maxGEN

    % Calculate the fitness value, i.e., the total distance of the path
    [fval, sumDistance, minPath, maxPath] = fitness(distances, pop);

    % Tournament selection
    tournamentSize = 4; % Set size
    for k = 1:popSize
```

```
        % Select parents for crossover
        tourPopDistances = zeros(tournamentSize, 1);
        for i = 1:tournamentSize
            randomRow = randi(popSize);
            tourPopDistances(i, 1) = sumDistance(randomRow, 1);
        end

        % Select the best, i.e., the one with the minimum distance
        parent1 = min(tourPopDistances);
        [parent1X, parent1Y] = find(sumDistance == parent1, 1, 'first');
        parent1Path = pop(parent1X(1, 1), :);

        for i = 1:tournamentSize
            randomRow = randi(popSize);
            tourPopDistances(i, 1) = sumDistance(randomRow, 1);
        end
        parent2 = min(tourPopDistances);
        [parent2X, parent2Y] = find(sumDistance == parent2, 1, 'first');
        parent2Path = pop(parent2X(1, 1), :);

        subPath = crossover(parent1Path, parent2Path, crossoverProbability); % Crossover
        subPath = mutate(subPath, mutationProbability); % Mutation

        offspring(k, :) = subPath(1, :);

        minPaths(gen, 1) = minPath;
    end
    fprintf('Generation:%d   Shortest Path:%.2fKM \n', gen, minPath);

    % Update
    pop = offspring;
    % Plot the shortest path under the current state
    if minPath < gbest
        gbest = minPath;
        paint(cities, pop, gbest, sumDistance, gen);

    end
end
figure
```

```
plot(minPaths, 'MarkerFaceColor', 'red', 'LineWidth', 1);
title('Convergence Curve (Shortest Path for Each Generation)');
set(gca, 'ytick', 500:100:5000);
ylabel('Path Length/km');
xlabel('Number of Iterations');

grid on
tEnd = toc(tStart);
fprintf('Time:%d minutes  %f seconds.\n', floor(tEnd / 60), rem(
    tEnd, 60));
```

Report on Use of AI

OpenAI ChatGPT(ChatGPT-3.5): We utilized ChatGPT as a tool to enhance the article by optimizing its grammar and sentence structure. We input the pre-written article into the ChatGPT model, requesting it to correct grammar errors, improve sentence structure, and suggest appropriate vocabulary replacements.