Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.079979

ARTICLE

Analysis of Metaheuristic, Sampling-Based, Potential Field, and Predictive Control Methods for Path Planning in Simulated Underwater Settings

Rubina Castro[1,2], **Bruno Silva**[1,3], **Luiz Guerreiro Lopes**[1,4] **and Fábio Mendonça**[1,2,*]

[1]Faculty of Exact Sciences and Engineering, University of Madeira, Funchal, Portugal
[2]Interactive Technologies Institute (ITI/LARSyS) and ARDITI, Funchal, Portugal
[3]Regional Secretariat for Education, Science and Technology, Regional Government of Madeira, Funchal, Portugal
[4]NOVA Laboratory for Computer Science and Informatics (NOVA LINCS), Caparica, Portugal

*Corresponding Author: Fábio Mendonça. Email: fabioruben@staff.uma.pt

Received: 01 February 2026; Accepted: 17 April 2026; Published: 15 June 2026

ABSTRACT: Path planning for autonomous underwater vehicles requires reliable and computationally efficient methods, particularly in cluttered environments. This work presents a comparative evaluation of representative approaches, including metaheuristic optimization methods (continuous genetic algorithm, particle swarm optimization, gray wolf optimizer, and Jaya), a sampling-based method (probabilistic roadmap with genetic refinement), a reactive strategy (artificial potential fields), and a control-based approach (model predictive control with control barrier functions). The algorithms are assessed in a controlled two-dimensional simulated workspace with randomly generated obstacles and systematically increasing obstacle density. Each configuration is evaluated across multiple independent trials using metrics such as success rate, path length, and convergence behavior. The effect of environmental disturbances is examined by analyzing particle swarm optimization under Gauss–Markov current models. The results show that performance depends strongly on the ability to preserve feasibility as obstacle density increases. The probabilistic roadmap with genetic refinement demonstrated the highest robustness, maintaining feasibility across all scenarios, while particle swarm optimization provided a strong balance between path quality and reliability in low-to-moderate clutter. The introduction of current disturbances led to reduced efficiency and consistency. Statistical analysis confirmed significant differences among methods, highlighting that rank-based superiority does not necessarily reflect practical robustness in constrained environments.

KEYWORDS: Autonomous underwater vehicles; path planning; collision avoidance; metaheuristic optimization; potential fields; sampling-based planning; predictive control

1 Introduction

Path-planning approaches are used in multiple fields, as they enable autonomous robots to navigate efficiently while avoiding obstacles and fulfilling mission objectives [1]. It is, therefore, the process through which autonomous robots determine an optimal trajectory that meets specific mission objectives, typically evaluated by metrics such as path distance, number of turns, and running time [2]. Depending on the level of environmental knowledge and real-time adaptability, path-planning methods can generally be divided into global and local approaches [3].

These path-planning capabilities are highly relevant for Autonomous Underwater Vehicles (AUVs) in deep-sea exploration, environmental monitoring, seabed mapping, and underwater inspection [4]. Due to the unpredictable and dynamic nature of marine environments characterized by ocean currents, communication delays, and complex obstacle distributions, path planning and trajectory optimization remain highly challenging tasks [5,6]. The goal is, therefore, to produce algorithms that can compute a collision-free and energy-efficient trajectory to allow the AUV to reach a target point safely while minimizing travel time, energy consumption, and deviation from the desired path [6].

Despite the substantial body of prior work, there remains a need for comparative analysis of controlled benchmarks in which algorithms are evaluated in the same controlled environment under varying obstacle densities and environmental disturbances. Furthermore, while many studies report path quality metrics such as length or smoothness, fewer works explicitly evaluate success rate, variability, and feasibility preservation as obstacle density increases. Such is the main goal of this work. The examined scenario is designed to enable simplified visualization of trajectories in a two-dimensional environment (which can be extended to higher dimensions). In addition, robustness to current-induced disturbances is investigated by considering simulated ocean-current conditions, including a constant-direction current and a continuously time-varying current.

The remainder of this work is organized as follows. Section 2 reviews representative AUV path-planning families. Section 3 describes the simulation environment and benchmarking methodology, including the evaluated approaches, obstacle-generation procedure, and performance metrics. Section 4 presents and discusses the results across obstacle densities and examines the impact of steady and time-varying current forcing on the algorithm performance. Section 5 concludes the paper by summarizing the main findings and discussing the key trade-offs observed.

2 Related Work

Path planning for AUVs sits at the intersection of motion planning, estimation, and control in operating conditions where uncertainty and vehicle and environmental constraints (e.g., bounded actuation, limited sensing, and ocean currents) are the norm [6]. Recent surveys emphasize that no single planner consistently dominates across “predictable” vs. “unpredictable” underwater environments [7,8]. As a result, practical solutions tend to be hybrid, pairing a global planner (for connectivity and long-horizon efficiency) with a local reactive layer or a short-horizon planning layer (for safety and rapid adaptation), often complemented by an optimization layer that accounts for dynamics, constraints, and mission costs [9].

Global path planning assumes prior knowledge of the environment and focuses on computing, before execution, an optimal or near-optimal path from the start to the goal with respect to a chosen cost model. Representative global methods include sampling-based techniques such as the Probabilistic Roadmap (PRM) [10,11] and population-based metaheuristics. The latter involve evolutionary methods such as Genetic Algorithms (GAs) [12,13], including their continuous-encoding variants (CGAs) [14]. Swarm-based methods such as Particle Swarm Optimization (PSO) [15,16] and the Gray Wolf Optimizer (GWO) [17,18], and metaphor-less optimization algorithms such as Jaya [19,20], all of which are used in this work. These methods search for feasible trajectories by optimizing cost functions related to path length, smoothness, and safety, often providing solutions for complex or high-dimensional environments.

In contrast, local path planning focuses on real-time navigation, where the robot dynamically reacts to changing environments and unforeseen obstacles. Potential field approaches such as the Artificial Potential Field (APF) method [21,22] provide reactive obstacle avoidance by combining attractive and repulsive terms, while Model Predictive Control (MPC) integrated with Control Barrier Functions (CBFs) [23,24] has shown good performance by continuously predicting future states and enforcing safety constraints

during motion execution. As a result, global and local approaches provide complementary capabilities: global algorithms emphasize route optimization, while local strategies enable adaptive obstacle avoidance and safe maneuvering in dynamic or partially unknown environments.

In recent years, numerous studies have focused on developing path-planning strategies adjusted specifically to AUV operations. Classical geometric and graph-based planners have been used due to their deterministic guarantees, although their scalability becomes limited in large or continuous underwater domains [25]. Sampling-based motion planners have also been adapted for AUV navigation, offering improved performance in high-dimensional spaces and uncertain environments [26,27]. In parallel, bio-inspired metaheuristic algorithms, such as PSO, GA, GWO, and Ant Colony Optimization (ACO), have gained notoriety for their flexibility and ability to escape local minima when navigating around complex obstacle fields or dynamic ocean currents [28,29].

More recently, control-theoretic approaches (such as integrating MPC with CBFs) have shown potential for safety-critical AUV navigation by ensuring constraint satisfaction and real-time collision avoidance in dynamic marine settings [23,30]. Together, these advances demonstrate the diversity and progression of underwater path planning, including sampling-based roadmaps, potential field methods, population-based metaheuristics, and constraint-based predictive control, each contributing complementary strengths. Accordingly, the remainder of this section focuses on these four families, with emphasis on navigation in clutter and under current induced disturbances.

In underwater settings, discretization and the cost model should reflect current-induced advection. Otherwise, nominally short routes can be far from time-efficient or energy-efficient, motivating current-adaptive cost formulations or frequent replanning [31,32]. For global connectivity in cluttered environments, sampling-based roadmaps such as PRM are often used to identify feasible corridors, which can then be refined by an optimizer to improve path quality [7,10].

Potential-field methods remain attractive for AUV path planning because they are simple, local, and computationally efficient, which makes them suitable for reactive obstacle avoidance. The canonical APF formulation combines an attractive term toward the goal with repulsive terms around obstacles [33], but it is susceptible to local minima and oscillations in cluttered geometries [34]. Consequently, APF variants for underwater navigation often add heuristics (e.g., intermediate targets, adaptive gains, or smoothing) or are coupled with higher-level guidance to mitigate trapping [35]. More recent work commonly integrates APF within metaheuristic or multi-objective optimization frameworks to improve global path quality and reduce failure modes associated with local minima [36,37].

Metaheuristic optimization has become widely used in AUV path planning because it can handle nonconvex, multi-criteria objectives (e.g., path length, smoothness, clearance, energy, and time) with relatively weak modeling requirements. GAs [14] and multi-objective evolutionary methods such as NSGA-II [38] are commonly used when a path is encoded as a sequence of intermediate control points and evaluated by a fitness function that aggregates costs and penalties. PSO [15] is widely adopted for continuous control-point parameterizations, while alternatives such as GWO [17] and metaphor-less schemes like Jaya [19] have been explored to reduce tuning effort. A recurring observation in surveys and comparative studies is that these methods can yield high-quality solutions in moderately cluttered environments [7,9], but feasibility and repeatability tend to deteriorate in narrow passages unless diversity preservation, effective constraint handling, or roadmap-based encodings are incorporated [32]. This sensitivity to environment structure motivates standardized benchmarking and reporting beyond mean path length, including success rate, computational time, and run-to-run variability, to enable fair comparisons across planning frameworks [39,40].

In line with these observations, control-point-based metaheuristics are sensitive to constraint handling and may stagnate in highly constrained geometries [9]. Roadmap-based planners explicitly encode connectivity and tend to be more robust when free space is fragmented [10], whereas MPC/CBF formulations prioritize constraint satisfaction over optimality [41,42].

Hybrid approaches have become a common design choice in underwater path planning: potential-field formulations provide fast, smooth, reactive guidance, while a higher-level optimizer such as PSO, GA, or ACO searches over control points, intermediate goals, or tuning parameters to improve global path quality and mitigate local-minima failure modes. Representative examples include the combination of APF and ACO for global planning with an added velocity-obstacle (VO)-based collision-avoidance layer for underwater vehicles [36], the explicit coupling of APF and PSO in a hybrid planner [43], and the use of PSO to optimize potential-field–driven navigation behavior in cluttered environments [44].

The same hybridization principle extends naturally to multi-AUV planning, where potential-field-like constructs can support coordination while an outer optimization layer addresses route quality and task-level objectives. Recent examples include dynamic velocity potential fields for cooperative hunting tasks [45] and improved multi-objective PSO for cooperative 3D planning in complex terrain and vortex-dominated currents [46]. This cooperative setting typically drives formulations toward explicitly multi-objective tradeoffs (e.g., efficiency, collision risk, cohesion) and, in many cases, decentralized or distributed implementations where each vehicle optimizes locally under shared constraints.

Ocean currents are a major driver of AUV-specific planning complexity because advection and spatiotemporal variability can dominate the effective motion and make purely geometric paths suboptimal (or even infeasible) [31]. Early work on evolutionary planning in variable oceans [12] and current–aware planning for gliders and AUVs showed that achievable improvements in energy consumption and transit time reductions depend strongly on the ratio between current speed and vehicle speed, motivating closer integration between planning algorithms and ocean models. Comparative studies of optimization-based planners under currents further emphasize that performance depends strongly on both the algorithm and the path encoding, and that robustness under time-varying currents commonly requires either replanning or cost formulations that account for currents [32].

A recurring conclusion in the literature (see, e.g., [31,32]) is that ocean-current effects should not be treated as a secondary consideration. Practical planners typically use a global model that accounts for currents, update plans frequently as state and current estimates evolve, or apply local feedback that compensates for drift without compromising obstacle avoidance. In parallel, control-based methods have gained visibility in AUV navigation because they offer a systematic way to enforce dynamic and safety constraints in real time. MPC optimizes a finite-horizon trajectory under explicit constraints [41], while CBFs provide formal safety conditions that can be imposed through quadratic programs or integrated into MPC [42]. Recent AUV-focused work combines these ideas in robust MPC formulations with fast-marching (FMM)-based travel time objectives and adaptive CBFs [23].

Compared with purely geometric planners, MPC and CBF approaches impose input and safety constraints via real-time optimization [42] and can be interpreted as optimization-based safety filters that minimally adjust a reference control objective while maintaining safety [47]. This makes them well-suited to respect actuator limits, vehicle footprint, and safety margins while responding to disturbances. However, in densely cluttered environments, they may become conservative or even reach deadlock, and they often require accurate models and real-time optimization [48]. For these reasons, surveys most often describe MPC and CBF as a local safety or tracking layer under a global planner (e.g., roadmaps or metaheuristics), rather than as a self-contained global planner [9].

Surveys also emphasize that high-performing AUV path-planning approaches are increasingly implemented as layered architectures that combine a global planner that explicitly encodes free-space connectivity (e.g., graph or sampling-based methods), an optimization step for path quality or multi-objective tradeoffs (e.g., PSO, GA, ACO, or NSGA-II), and a control layer that accounts for both dynamics and safety, often implemented with MPC and CBFs [7,9]. Key remaining gaps include standardized, reproducible benchmarks that incorporate realistic sensing and current dynamics, where reusable underwater simulation frameworks can play an important role [49,50].

Evaluation practice is also improved by reporting metrics beyond mean path length, such as success rate and performance variability across runs [51]. Evidence of robustness when moving from simulation to open-water trials remains limited, motivating more systematic validation through approaches such as hardware-in-the-loop testing [52].

A recent work by Yin and Xiang [53] proposes an adaptive differential evolution-based multi-source information fusion framework for real-time cooperative navigation of multiple unmanned surface vehicles in dynamic maritime environments. Their approach integrates heterogeneous sources of information, including reciprocal velocity obstacles and goal-alignment cues, with fusion weights optimized via adaptive differential evolution. Unlike traditional optimization-based planners that rely on static assumptions or reinforcement learning methods requiring extensive training, this method achieves real-time decision-making through a training-free mechanism with adaptive parameter tuning. Such methodology can be a suitable next step for this work to integrate multi-agent coordination.

It is, however, important to note that Yin and Xiang [53] primarily target cooperative navigation systems with explicit inter-agent communication and interaction modeling. In contrast, the present work focuses on benchmarking path planning strategies under controlled environmental disturbances and emphasizes comparative evaluation across different planning paradigms.

3 Materials and Methods

To ensure reproducibility and enable a controlled comparison, the computing platform and simulation environment are first described. The evaluated approaches are then summarized, followed by the experimental setup, performance metrics, and common parameter settings used throughout the study.

3.1 Computing Platform and Simulation Environment

To reflect resource constraints typical of onboard embedded AUV deployments, all simulations were executed on an embedded-class computing testbed (NVIDIA Jetson AGX Orin 64 GB) using Julia (v1.10.4). A two-dimensional workspace was adopted to facilitate clear trajectory visualization while preserving the core geometric challenges of obstacle avoidance.

The simulated two-dimensional workspace is defined as $\Omega = [0, 20] \times [0, 20] \subset \mathbb{R}^2$, with start and goal positions $\mathbf{x}_s = [0, 0]$, $\mathbf{x}_g = [20, 20]$, respectively. Circular obstacles, representing cross-sectional approximations of underwater rock formations, were generated with random centers in Ω and radii $r \in [0.5, 2.0]$. Ocean rocks often exhibit irregular, nonuniform geometries, and the partial overlap of circular obstacles in the simulated environment can produce composite shapes that resemble these random natural formations. Obstacle fields were regenerated between independent runs to ensure statistical variability. Environmental complexity was increased by varying the number of obstacles according to $n_{\text{obs}} \in \{0, 2, 4, 6, 8, 10, 20, 30, 40\}$. A margin of 1 unit was imposed on the border of each dimension to avoid collision between the AUV and the walls.

During scenario generation, explicit geometric constraints were imposed to prevent obstacles from overlapping the start and goal regions. For each obstacle, with obstacle margin O_M, center $\mathbf{c}$, and radius r, the conditions $\|\mathbf{c} - \mathbf{x}_s\| > r + O_M$ and $\|\mathbf{c} - \mathbf{x}_g\| > r + O_M$ were enforced. In addition, a minimum center-to-center separation of $0.55(r_i + r_j)$ was adopted between obstacles, allowing for a more controlled partial overlap and thus the formation of irregular composite shapes. To avoid trivial blockage near the initial position, a preliminary feasibility check requires at least two free outgoing directions from the start region.

The theoretical Euclidean distance between the nominal start point (0.0, 0.0) and the goal (20.0, 20.0) is 28.2842712475. However, some reported average path lengths are slightly lower because the simulations do not initialize all particles exactly at (0, 0). Instead, the initial positions are generated by a function with a lower bound of [0.0] and an upper bound of [0.35, 0.35], corresponding to the sensor radius, and placing the particles within a small bounded region around the nominal start. Consequently, some trajectories begin from points that are marginally closer to the goal, leading to traveled distances below the theoretical straight-line distance defined from the exact start point.

Finally, all scenarios were subjected to an A*-based path feasibility validation during the scenario generation stage. A* validates each scenario by treating the obstacle field as a maze-like environment, where the free space corresponds to traversable corridors and the obstacles define blocked regions. In this representation, the algorithm searches for a valid route from the start to the goal in the same way one would solve a maze, progressing only through admissible passages while avoiding occupied areas. If A* is able to connect the start and goal through the available free space, the scenario is considered feasible. Otherwise, it is rejected as non-navigable. In this sense, the method acts as a maze solver that verifies whether the obstacle arrangement still leaves an open corridor for motion. This idea is useful because it ensures that each tested scenario contains at least one possible solution, so that later failures of the evaluated algorithms reflect their own search or control limitations rather than an impossible environment.

For each value of n_{obs}, except 0, a total of 31 valid scenarios were generated and shared across all evaluated algorithms, thereby ensuring a consistent and controlled comparison under identical environmental conditions. Therefore, all planning algorithms operate under the assumption of a fully known static environment, where the obstacle map is available a priori. This allows the evaluation to focus on planning performance rather than perception or mapping uncertainty.

3.2 Evaluated Approaches

A representative set of approaches was selected to cover metaheuristic optimization, classical reactive navigation, sampling-based planning, and optimization-based predictive control. Specifically, APF was included as a potential-field (reactive) baseline [21,22].

Evolutionary computation was covered by CGA [14]. Swarm-intelligence optimization was represented by PSO [15,16] and GWO [54], while Jaya was included as a metaphor-less, parameter-free optimizer [19,20]. Sampling-based motion planning was implemented using PRM [10,11] while safety-oriented predictive control was represented by MPC with CBF [23,24]. The implemented algorithms of these last two approaches are presented in Appendix A, as they were custom implementations (the other used algorithms were implemented with procedures that are standard in state-of-the-art). As commonly observed in practice, no single method is expected to dominate across all operating conditions [55]. This motivates the use of comparative benchmarks conducted under uniform conditions and reported using consistent metrics.

3.3 Benchmark Setup and Evaluation Procedure

Population-based methods used a population size of 50. Each algorithm configuration pair was evaluated over 31 independent runs with a maximum of 300 iterations each. For CGA and PRM-based optimization, a limit of 300 generations was used. For the MPC-CBF method, the prediction horizon was fixed at 35, the maximum control input was limited to 1, the workspace boundaries were defined as −1 and 21, the control barrier function safety margin was set to 0.05, and the sampling interval was 0.3. For the PRM-GA planner, the PRM was generated with 100 sampled nodes and 8 nearest-neighbor edges, while the GA refinement employed 8 waypoints and 40 generations. These parameter values were used consistently throughout the experiments to enable fair and computationally manageable comparisons among the evaluated methods. In the proposed framework, the initial motion update was designed to balance exploitation and exploration.

For PSO, the initial particle velocities were defined as a combination of a goal-directed component and a stochastic exploration term. The goal-directed component introduces an initial bias toward the target, scaled by a factor of 0.4, which promotes early convergence without overly constraining the search. The stochastic component was generated from a uniform distribution and centered by subtracting 0.5, ensuring symmetric perturbations with zero mean. Its magnitude was scaled by 0.08, resulting in small but non-negligible variability in the initial particle movements. This initialization strategy provides a balance between exploitation and exploration at the start of the optimization process.

Furthermore, to assess the effect of environmental disturbances, two Gauss–Markov current models were adopted to represent temporally correlated environmental disturbances during navigation, with a moderate (MOD) current level and a strong (STR) current level [56,57]. In this configuration, the current was parameterized with a mean flow component in the two horizontal directions set within the range 0.2–0.3, representing a persistent but moderate drift. The corresponding stochastic variations were limited by standard-deviation terms of 0.03 and correlation-related parameters of 0.06, producing smooth fluctuations rather than abrupt changes in the flow field.

A scaling factor of 0.5 was applied to control the overall magnitude of the MOD current disturbance. This setting was selected to emulate a moderate underwater current, allowing the evaluation of path-planning robustness under realistic, time-correlated environmental forcing. To emulate a stronger underwater disturbance, a similar Gauss–Markov current process was defined with a scaling factor of 1.2 (for STR current), representing a persistent strong drift. Compared with the moderate-current setting, this parameterization introduces substantially larger mean current components in both horizontal directions, thereby exerting a stronger influence on the vehicle trajectory. Hence, the stochastic terms remain relatively small, ensuring smooth temporal variation, while the increased correlation coefficient yields a more persistent flow pattern. This strong-current scenario provides a demanding test condition for evaluating the resilience of the path-planning algorithms in the presence of significant time-correlated environmental forcing.

In all simulations, a goal-reaching threshold G_T of 0.01 was used, which means that the target was considered achieved only when the agent was within 0.01 units of the goal position. Furthermore, O_M was set to 0.2 to ensure a minimum safety clearance during navigation and collision checks. These parameters support accurate path termination and safer obstacle avoidance throughout the simulated trajectories. To ensure the existence of a free passage between nearby obstacles, a minimum separation distance was imposed during scenario generation. Given that each obstacle was expanded (with O_M for collision checking), the clearance between obstacle boundaries had to be at least 0.4. This condition preserves a feasible corridor for agent motion after accounting for the safety margins associated with neighboring obstacles. For reproducibility, the pseudo-random number generator was initialized with seed 1234, ensuring that the stochastic components of the experiments remain consistent across repeated runs.

The fitness function was designed to jointly promote goal-directed progress, directional consistency, and obstacle-safe motion while discouraging stagnation and cyclic behavior. For a candidate position, the main objective term is the Euclidean distance to the goal. This term is augmented by a stagnation penalty that increases when the displacement from the current position is negligible, thereby discouraging extremely short or null movements. A directional alignment penalty is also introduced based on the dot product between the candidate motion direction and the goal direction, so that movements poorly aligned with the target trajectory are penalized. To explicitly reward effective advancement, a progress term is included by comparing the current distance-to-goal with the candidate's distance-to-goal. Goal-approaching moves reduce the fitness value, whereas backward or non-progressive moves are penalized more strongly.

In addition, a memory-based penalty is applied to prevent the particle from returning to its previous location or reversing the direction of the last step, which helps reduce oscillatory and backtracking behavior. Finally, an obstacle-clearance penalty is imposed according to the distance between the candidate position and each obstacle boundary, with increasingly large penalties assigned as the candidate approaches or enters obstacle-inflated safety regions. Hence, the proposed fitness function balances convergence efficiency, motion smoothness, local exploration control, and collision avoidance within a unified evaluation criterion.

For each configuration, the number of iterations required to reach the goal and the total path length were recorded for every run. Results are summarized using the mean, standard deviation, and range (minimum and maximum) to capture both typical performance and run-to-run variability.

4 Results and Discussion

This section reports and interprets the simulation results obtained under the benchmark described in Section 3. It starts with an algorithmic benchmark and then further examines the results with statistical analysis.

4.1 Algorithmic Benchmark

Representative trajectories for each algorithm, except PSO, are shown in Fig. 1. PSO was then subjected to a deeper analysis in scenarios involving current. Thus, representative PSO trajectories for the baseline model, Gauss–Markov MOD current, and Gauss–Markov STR current cases are shown in Fig. 2a–c, respectively. PSO was selected for this subsequent analysis because its trajectory generation is directly influenced by velocity updates, making it particularly suitable for analyzing the impact of external perturbations such as ocean currents. This allows the disturbance effects to be isolated and interpreted in a controlled manner, while keeping the computational scope of the study manageable.

The representative trajectories figures show how the algorithms can traverse the space and follow paths that are between the obstacles, usually with very smooth curves. Quantitative results for all algorithms are shown in Tables 1 and 2, and the following subsections examine them.

Fig. 3 shows an example when the algorithms were not successful in reaching the goal. For APF (Fig. 3a), the agent behaves like it is trapped in a local minimum formed by the combined repulsive fields of nearby obstacles, oscillating indefinitely in a region from which no gradient descent on the potential surface leads toward the goal. This entrapment is a well-documented limitation of the canonical APF formulation, and it becomes increasingly frequent as obstacle density rises, since the probability of forming enclosed repulsive basins grows with the number of obstacles. In the CGA failure case (Fig. 3b), the chromosome population converges prematurely toward a suboptimal region of the search space. Once the genetic diversity is exhausted, the algorithm lacks the variability needed to redirect the encoded path around the blocking configuration, and the resulting trajectory stagnates without reaching the goal. The PSO baseline failure

(Fig. 3c) presents a similar swarm-collapse phenomenon, in which particles cluster around a locally attractive but infeasible region, and the cognitive and social acceleration terms are insufficient to pull the swarm out of a narrow passage that offers no direct progress toward the goal.

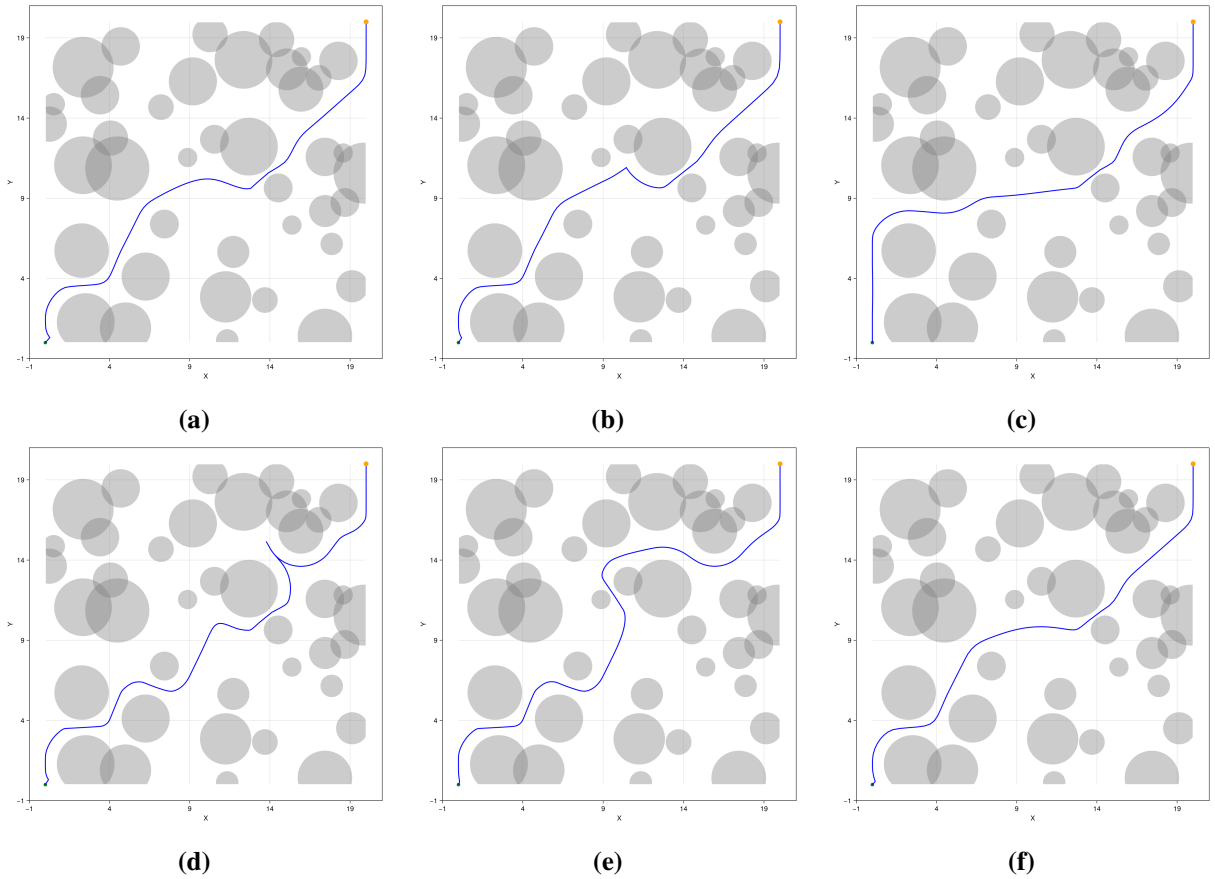

Figure 1: Representative trajectories in cluttered environments with 40 obstacles for (**a**) APF, (**b**) CGA, (**c**) PRM–GA, (**d**) GWO, (**e**) Jaya, and (**f**) MPC–CBF.

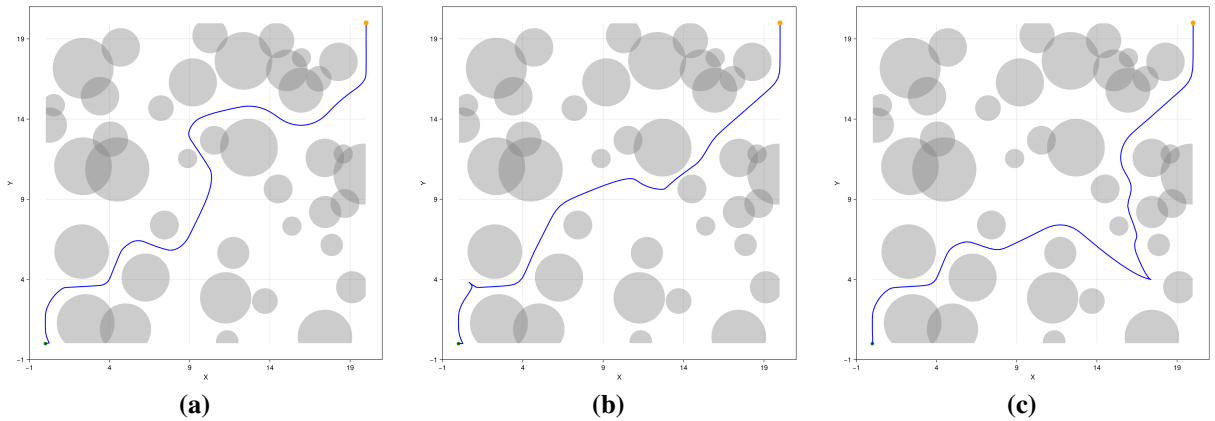

Figure 2: Representative trajectories in cluttered environments with 40 obstacles for PSO (**a**) baseline, (**b**) with Gauss-Markov moderated current, and (**c**) with Gauss-Markov strong current.

Table 1: Consolidated path-length summary statistics across obstacle densities for all evaluated approaches. Each entry reports Mean ± StdDev on the first line and [Min, Max] on the second line.

n_{obs}	APF	CGA	GWO	JAYA	MPC–CBF	PRM–GA	PSO	PSO Gauss–Markov MOD current	PSO Gauss–Markov STR current
0	28.0229954479 ± 0.1042464666 [27.8127039852, 28.1567395440]	27.8335120962 ± 0.0241352379 [27.7989854469, 27.8926902895]	27.9259945580 ± 0.0743637698 [27.7972184146, 28.0000008956]	27.9255066960 ± 0.0646758628 [27.7980716319, 28.0000008677]	28.0605522168 ± 0.0962139140 [27.8905979358, 28.2533767730]	28.2842712475 ± 0.0000000000 [28.2842712475, 28.2842712475]	27.9098161290 ± 0.0748294421 [27.7960000000, 28.0000000000]	29.2923387097 ± 1.2718955410 [28.0468000000, 33.4490000000]	30.3683838710 ± 2.3527501260 [28.2389000000, 39.9618000000]
2	28.3581155825 ± 0.6774213788 [27.8705127021, 30.9668306748]	28.0067104740 ± 0.3780619876 [27.7906813025, 29.4940899752]	28.3169148505 ± 0.7369939003 [27.7988126569, 30.5108584040]	28.3548836015 ± 0.7592165984 [27.7997809461, 30.6996730677]	28.2577533368 ± 0.4399780968 [27.8882243586, 29.7491098531]	28.3750390748 ± 0.1845805662 [28.2842712475, 29.0596064338]	28.3650677419 ± 0.7394394194 [27.8000000000, 30.5403000000]	29.5599677419 ± 1.0694124843 [28.2727000000, 32.2575000000]	30.1719758621 ± 1.8277989215 [27.9896000000, 36.2408000000]
4	30.5722914973 ± 6.3233425281 [27.9179736378, 59.8969946420]	28.5825298057 ± 1.7367861131 [27.8042423453, 36.4339726857]	28.9506444367 ± 1.2776769584 [27.7926936041, 32.1277519540]	28.9454385430 ± 1.2392590136 [27.8207697627, 32.0183142287]	28.7532546216 ± 1.1604137237 [27.8683973601, 33.4025686134]	28.6500196129 ± 0.7280858823 [28.2842712475, 31.9605551434]	28.9616233333 ± 1.2712300822 [27.8000000000, 32.1367000000]	29.9718700000 ± 2.0362887689 [28.3947000000, 36.6422000000]	30.5509222222 ± 2.1090913611 [28.3880000000, 37.7894000000]
6	28.4956734471 ± 0.6759026175 [27.8455972107, 30.4186292517]	28.0826878038 ± 0.3616309939 [27.7985228093, 29.4104984892]	28.6614919749 ± 0.8853783809 [27.8000002252, 30.5985848473]	28.6403142964 ± 0.8676413273 [27.7953912641, 30.5096551590]	28.3828752958 ± 0.4632725619 [27.9211282180, 29.8103336429]	28.4362597719 ± 0.3041100055 [28.2842712475, 29.8112182911]	28.6533354839 ± 0.8633916738 [27.7984000000, 30.5767000000]	29.5983354839 ± 1.5327100921 [28.0582000000, 34.9485000000]	29.7254310345 ± 1.6523596148 [28.1264000000, 36.4918000000]
8	29.9754646033 ± 2.1211614920 [27.8650745776, 34.9893511021]	29.1238234615 ± 2.3479954175 [27.7985654238, 39.9784187666]	29.7767653230 ± 1.4901807279 [27.8900259695, 32.3723739090]	29.7710616484 ± 1.5543918959 [27.7976405788, 32.5553457538]	29.1997366966 ± 1.0876244183 [27.8514105715, 31.8473239703]	28.7814964217 ± 0.5728468257 [28.2842712475, 30.6047289153]	29.7550548387 ± 1.5466779813 [27.7964000000, 32.5779000000]	30.4364758621 ± 2.3085434161 [28.1769000000, 38.6831000000]	31.0232161290 ± 3.1204915313 [28.2499000000, 44.7952000000]
10	32.2563057253 ± 10.1044350429 [27.9323234437, 84.5139419149]	32.9891621181 ± 19.4427688482 [27.8041100786, 134.9580385543]	30.0031447792 ± 1.5146619819 [27.7957801221, 33.0805798456]	29.9915703842 ± 1.5491074133 [27.7963164487, 33.2556840072]	29.2552803101 ± 1.0355589439 [27.9591795289, 31.1568745822]	28.9377168371 ± 0.7666896493 [28.2842712475, 31.1947575877]	30.0015166667 ± 1.5413584705 [27.8000000000, 33.0704000000]	29.9763444444 ± 1.5861197923 [28.1574000000, 33.8527000000]	31.2657821429 ± 3.5117320937 [28.1571000000, 42.6408000000]
20	34.6857593303 ± 6.0745858358 [28.0609064151, 49.2589958958]	31.7979662799 ± 5.1670682087 [27.9523079082, 45.7797018248]	31.9442496059 ± 1.9355676777 [27.9681044254, 36.1604811702]	31.9298645759 ± 1.9161625408 [28.1999998567, 35.7140862905]	30.0715891344 ± 1.2905800798 [28.2652389285, 33.5810408999]	30.1995381666 ± 1.6441030019 [28.2988604717, 34.4308290276]	31.8539034483 ± 1.8731417347 [28.1721000000, 35.8098000000]	32.6722913043 ± 3.7947570589 [29.2558000000, 43.8165000000]	32.9120545455 ± 4.3432125926 [28.7943000000, 47.9529000000]
30	40.1336195917 ± 13.3893227519 [29.0978035433, 88.7681420951]	43.5530835238 ± 14.6205016355 [28.6458009337, 92.5093753658]	33.9154585705 ± 2.2382085694 [30.1084158122, 39.4268824067]	34.0727945631 ± 2.3151850839 [30.0985752639, 39.5056381405]	31.0124221706 ± 1.4322658715 [28.7381942408, 34.3976404391]	31.2065386265 ± 2.0786907447 [28.7028764442, 36.7868636743]	34.7471250000 ± 3.7464755815 [29.7388000000, 48.6142000000]	33.5941166667 ± 3.1085559110 [29.9150000000, 41.3477000000]	35.3107500000 ± 5.7803626186 [29.5760000000, 51.0625000000]
40	53.5107021893 ± 14.0171480498 [30.6682465712, 72.8103643154]	55.8580550951 ± 29.6195833675 [29.2576637714, 138.7698973041]	36.9230565817 ± 4.2772490484 [31.5814284000, 44.2696746972]	36.6694581428 ± 4.1853404750 [31.5739963820, 44.2877661861]	32.0092050968 ± 1.6745792237 [29.3769472001, 35.1100737069]	32.6268905203 ± 2.0545357290 [29.4615123114, 37.0326799666]	36.8081666667 ± 3.9289532934 [31.5775000000, 44.5724000000]	32.7218250000 ± 2.4278217449 [30.6344000000, 40.9919000000]	37.0498333333 ± 3.2275000778 [29.6932000000, 36.8361000000]

Table 2: Number of successful and failed occurrences, indicated as (success/failure) counts, for each algorithm across obstacle densities.

Algorithm	0	2	4	6	8	10	20	30	40
APF	31/0	31/0	31/0	31/0	29/2	31/0	26/5	22/9	14/17
CGA	31/0	31/0	31/0	31/0	31/0	30/1	28/3	26/5	24/7
GWO	31/0	31/0	30/1	31/0	31/0	30/1	29/2	23/8	18/13
JAYA	31/0	31/0	30/1	31/0	31/0	30/1	29/2	23/8	19/12
MPC–CBF	31/0	31/0	31/0	31/0	31/0	30/1	26/5	19/12	16/15
PRM–GA	31/0	31/0	31/0	31/0	31/0	31/0	31/0	31/0	31/0
PSO	31/0	31/0	30/1	31/0	31/0	30/1	29/2	24/7	21/10
PSO–GM MOD	31/0	31/0	30/1	31/0	29/2	27/4	23/8	18/13	9/22
PSO–GM STR	31/0	29/2	27/4	29/2	31/0	28/3	22/9	16/15	8/23

In both the GWO (Fig. 3d) and Jaya (Fig. 3e) cases, the failure mechanism is closely analogous, as it is likely due to the population-level update rules that drive candidate positions toward the current best estimates reinforce convergence to a region that is geometrically cut off from the goal, and the absence of an explicit connectivity representation means the search has no systematic mechanism to detect or escape this topological blockage.

Finally, the MPC–CBF failure (Fig. 3f) arises from a different source. The local optimizer responsible for computing the next control action becomes infeasible when the safety constraints imposed by multiple nearby obstacles simultaneously restrict the feasible input set to an empty region, or when the tangent-based fallback maneuver is also blocked. Thus, unlike the population-based methods, which fail due to search stagnation, MPC–CBF fails due to constraint infeasibility in dense configurations, consistently with the known trade-off between safety guarantees and completeness in reactive constrained control.

Across all methods, it is noteworthy that the failure trajectories shown here represent scenarios in which a feasible path does exist (as verified by the A* pre-screening), confirming that the observed failures are attributable to the algorithmic limitations of each method rather than to the environment being inherently unsolvable.

Although the trajectories shown in Fig. 3 represent individual illustrative instances, they are consistent with the general failure behavior observed across the full set of tested scenarios at high obstacle densities. APF and the population-based methods repeatedly exhibited entrapment in local minima or premature convergence, while MPC–CBF failures were consistently associated with constraint infeasibility in narrow corridors. However, PRM–GA produced no failure case across any of the tested configurations, a result that can be attributed to its fundamentally different planning paradigm, that explicitly constructs a collision-free roadmap of the free space prior to optimization, thus, it allows the method to encode global connectivity as a structural guarantee rather than an emergent property of iterative search, so that the genetic refinement stage operates exclusively over paths that are already known to connect the start and goal regions. This explicit topological encoding makes PRM–GA robust to the geometric complexity that defeats reactive and population-based methods.

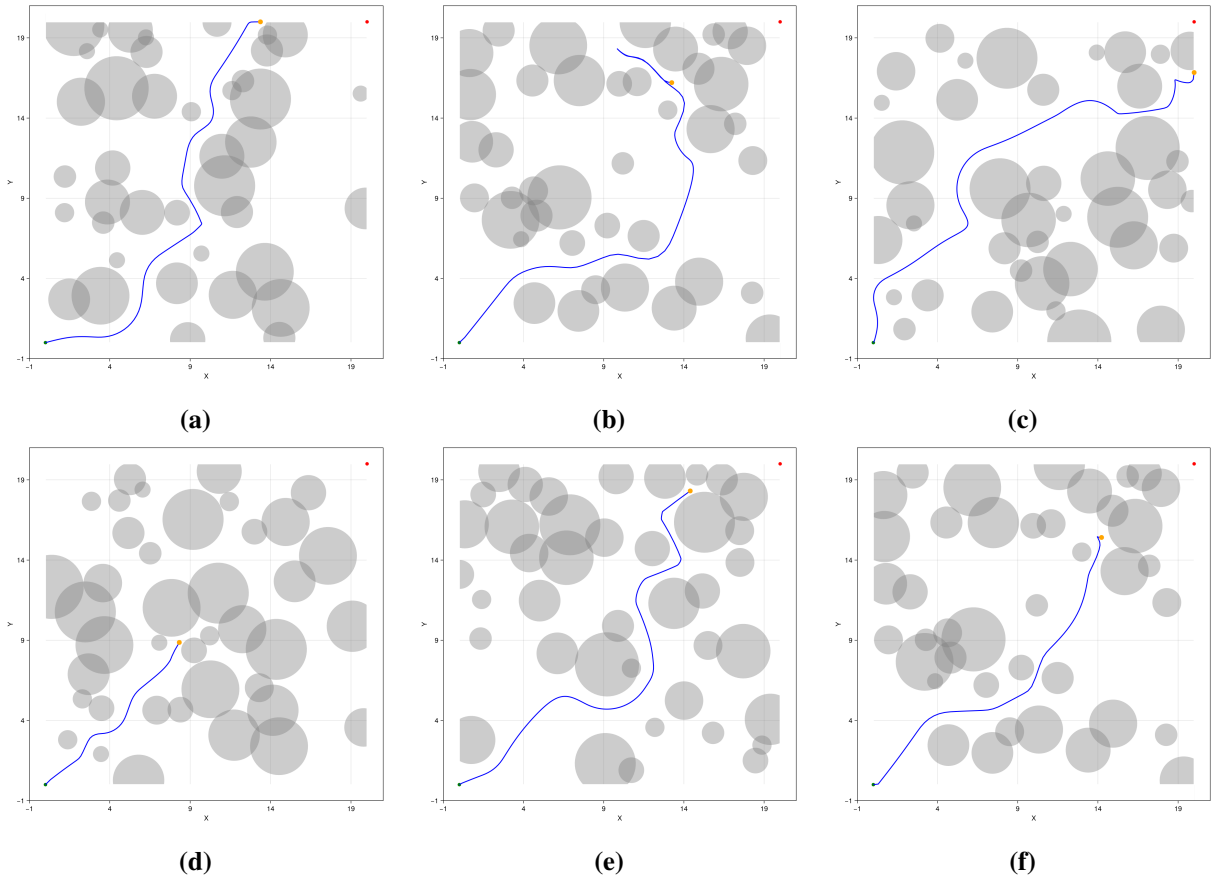

Figure 3: Representative failure trajectories in cluttered environments with 40 obstacles for (**a**) APF, (**b**) CGA, (**c**) PSO baseline, (**d**) GWO, (**e**) Jaya, and (**f**) MPC–CBF.

4.1.1 Artificial Potential Field

In the obstacle-free case, APF produced a mean path length of 28.023 with a very small standard deviation of 0.104, and all 31 runs were successful, demonstrating stable behavior in an unconstrained environment. Under low clutter levels, the method remained reasonably effective, yielding mean path lengths of 28.358, 30.572, 28.496, and 29.975 for 2, 4, 6, and 8 obstacles, respectively. Nevertheless, dispersion increased noticeably in some scenarios, particularly at 4 obstacles, where the standard deviation reached 6.323, and the maximum path length rose to 59.897, indicating that although many runs remained efficient, some trajectories became substantially longer due to unstable local interactions with obstacles.

As obstacle density increased, both efficiency and robustness deteriorated. At 10 obstacles, the mean path length rose to 32.256, while the standard deviation increased sharply to 10.104, and the maximum reached 84.514, revealing pronounced run-to-run variability even though all 31 runs still returned valid trajectories. Beyond this point, reliability also began to decline. At 20 obstacles, the mean path length increased to 34.686, with 26 successful runs. In denser environments, the degradation became more severe. At 30 obstacles, APF achieved a mean of 40.134 with a standard deviation of 13.389, but only 22 runs were successful. At 40 obstacles, the mean path length further increased to 53.511 with a standard deviation of 14.017, while the success count dropped to 14. This combined increase in average path length, variability, and failure frequency indicates that APF becomes progressively less reliable in highly cluttered maps.

Overall, the APF results are consistent with the well-known limitations of potential-field-based planning in complex environments. Although the method performs well in simple or moderately constrained settings, dense obstacle distributions intensify oscillatory motion near repulsive regions, increase sensitivity to geometric layout, and raise the likelihood of local minima entrapment. These effects jointly reduce path quality and compromise robustness, explaining the marked deterioration observed at high obstacle densities.

4.1.2 Continuous Genetic Algorithm

For the obstacle-free case, CGA achieved a mean path length of 27.834 with a very small standard deviation of 0.024, and all 31 runs were successful, indicating highly stable behavior in an unconstrained environment. Under low clutter levels, the algorithm remained effective, with mean path lengths of 28.007, 28.583, 28.083, and 29.124 for 2, 4, 6, and 8 obstacles, respectively. Although the average performance remained competitive in these scenarios, variability increased with obstacle density, particularly at 4 obstacles, where the standard deviation rose to 1.737, and the maximum path length reached 36.434, and at 8 obstacles, where the standard deviation further increased to 2.348, and the maximum path length rose to 39.978. Nevertheless, all runs remained successful up to 8 obstacles, suggesting that CGA maintained robustness across sparse and moderately cluttered maps.

As obstacle density increased further, the behavior became substantially less stable. At 10 obstacles, the mean path length rose to 32.989, accompanied by a large standard deviation of 19.443 and a maximum of 134.958, while the success count decreased to 30. This result indicates that, although the algorithm often still found feasible paths, some runs became highly inefficient. At 20 obstacles, the mean decreased slightly to 31.798, but variability remained elevated, with a standard deviation of 5.167 and 3 failed runs. In denser environments, however, performance degradation became much more pronounced. At 30 obstacles, the mean path length increased to 43.553 with a standard deviation of 14.621, while the number of successful runs dropped to 26. At 40 obstacles, CGA exhibited its worst overall performance, with a mean path length of 55.858, a very large standard deviation of 29.620, a maximum of 138.770, and only 24 successful runs. These results indicate a marked loss of both efficiency and robustness in heavily cluttered environments.

Overall, the CGA results suggest that the method is capable of producing short and highly consistent trajectories in simple and moderately constrained scenarios, but its stability deteriorates as obstacle density increases. The growth in variability, the occurrence of very long trajectories, and the rise in failed runs at higher clutter levels suggest that the algorithm becomes increasingly sensitive to the complexity of the search space. In dense maps, this behavior is likely associated with more difficult convergence, less effective obstacle circumvention, and a greater tendency to explore inefficient regions before reaching the goal, ultimately reducing both path quality and reliability.

4.1.3 Probabilistic Roadmap–Genetic Algorithm

In the obstacle-free case, PRM–GA produced a mean path length of 28.284 with zero standard deviation, and all 31 runs were successful, indicating ideal, consistent behavior in a simple environment. Under low and moderate clutter, the method remained efficient and highly robust, yielding mean path lengths of 28.375, 28.650, 28.436, 28.781, and 28.938 for 2, 4, 6, 8, and 10 obstacles, respectively. In all these cases, all 31 runs were successful. Variability also remained low, with standard deviations between 0.185 and 0.766, and the maximum path lengths stayed comparatively close to the corresponding means, indicating stable and repeatable planning performance across sparse and moderately cluttered scenarios.

As obstacle density increased, PRM–GA continued to exhibit strong robustness. At 20 obstacles, the mean path length rose to 30.200 with a standard deviation of 1.644, while all 31 runs still succeeded. At 30

obstacles, the mean increased only slightly further to 31.207, with a standard deviation of 2.079, and again, no failed runs. Even in the densest case, with 40 obstacles, PRM–GA maintained full reliability, achieving 31 successful runs, with a mean path length of 32.627, a standard deviation of 2.055, and a maximum of 37.033. Although path length increased gradually with obstacle density, the growth remained controlled compared with several other methods, and the algorithm preserved both feasibility and consistency throughout all tested environments.

Overall, the PRM–GA results indicate a highly robust planning strategy across the full range of obstacle densities considered. The combination of consistently low dispersion, moderate growth in mean path length, and an ideal success rate in all scenarios suggests that the roadmap-based representation provides reliable global connectivity, while the genetic refinement stage helps improve path quality without compromising feasibility.

4.1.4 Gray Wolf Optimizer

Regarding the obstacle-free case, GWO achieved a mean path length of 27.926 with a standard deviation of 0.074, and all 31 runs were successful, indicating stable performance in an unconstrained environment. Under low and moderate clutter levels, the algorithm maintained relatively consistent behavior, with mean path lengths of 28.317, 28.951, 28.661, and 29.777 for 2, 4, 6, and 8 obstacles, respectively. Variability remained moderate in these cases, with standard deviations ranging from 0.737 to 1.490. However, robustness was already slightly affected at 4 obstacles, where 30 out of 31 runs were successful.

As obstacle density increased further, the degradation became more evident. At 10 obstacles, the mean path length rose to 30.003, with a standard deviation of 1.515, and the number of successful runs decreased to 30. At 20 obstacles, GWO produced a mean path length of 31.944 and a standard deviation of 1.936, with 29 successful runs. In denser maps, both path efficiency and robustness deteriorated more substantially. At 30 obstacles, the mean path length increased to 33.915 with a standard deviation of 2.238, while the success count dropped to 23. At 40 obstacles, the mean path length further increased to 36.923, with a standard deviation of 4.277, and only 18 runs were successful. The corresponding maximum value of 44.270 at 40 obstacles also indicates that some successful runs required considerably longer detours in the most cluttered environments.

Therefore, the GWO results suggest that the algorithm performs well in simple and moderately cluttered scenarios, where it yields short trajectories with relatively low variability. However, its robustness declines progressively as obstacle density increases, as reflected by the growing number of failed runs in dense maps. This pattern indicates that, although GWO can still produce competitive solutions when feasible paths are easier to exploit, its search dynamics become less reliable in highly cluttered environments, where the optimization process is more prone to premature convergence, difficulty in escaping unfavorable regions, or reduced ability to maintain feasible path evolution. Consequently, GWO appears suitable for low-to-moderate obstacle densities, but less robust than the strongest-performing methods in densely obstructed settings.

4.1.5 Jaya

In the obstacle-free case, JAYA achieved a mean path length of 27.926 with a standard deviation of 0.065, and all 31 runs were successful, indicating stable and highly consistent behavior in an unconstrained environment. Under low and moderate clutter, the algorithm maintained similarly competitive performance, with mean path lengths of 28.355, 28.945, 28.640, and 29.771 for 2, 4, 6, and 8 obstacles, respectively. Variability remained moderate in these scenarios, with standard deviations ranging from 0.759 to 1.554.

With further increases in obstacle density, the degradation became progressively more apparent. At 10 obstacles, the mean path length rose to 29.992, with a standard deviation of 1.549, and the number of successful runs decreased to 30. At 20 obstacles, JAYA produced a mean path length of 31.930 and a standard deviation of 1.916, with 2 failures. In denser environments, efficiency and robustness declined substantially. At 30 obstacles, the mean path length increased to 34.073, with a standard deviation of 2.315, while the success count fell to 23. At 40 obstacles, the mean path length further increased to 36.669, with a standard deviation of 4.185, and 12 failures. The maximum value of 44.288 at 40 obstacles also indicates that some successful runs involved noticeably longer detours in the most cluttered maps.

The JAYA results indicate that the algorithm performs well in simple and moderately cluttered environments, since it can produce short paths with relatively low dispersion. However, as with other population-based methods, its robustness decreases as obstacle density increases, as evidenced by the increasing number of failures and the gradual rise in both mean path length and variability. These results suggest that, although JAYA is effective at exploring feasible solutions in lower-complexity scenarios, its search mechanism becomes less reliable in highly constrained maps, where feasible corridors are narrower, and the optimization process is more sensitive to local geometric complexity.

4.1.6 Model Predictive Control Integrated with Control Barrier Functions

For the obstacle-free case, MPC–CBF achieved a mean path length of 28.061 with a standard deviation of 0.096, and all 31 runs were successful, indicating stable and highly repeatable behavior in an unconstrained environment. Under low and moderate clutter levels, the method remained both efficient and consistent, producing mean path lengths of 28.258, 28.753, 28.383, and 29.200 for 2, 4, 6, and 8 obstacles, respectively. Variability remained comparatively low in these scenarios, with standard deviations ranging from 0.440 to 1.160, and all 31 runs were successful up to 8 obstacles, confirming strong robustness across sparse and moderately cluttered maps.

As obstacle density increased further, the performance degradation remained gradual at first. At 10 obstacles, MPC–CBF produced a mean path length of 29.255 with a standard deviation of 1.036, while 30 runs were successful. At 20 obstacles, the mean increased to 30.072, with a standard deviation of 1.291, but the success count dropped more noticeably to 26. In denser environments, reliability deteriorated further. At 30 obstacles, the mean path length reached 31.012 with a standard deviation of 1.432, while only 19 runs were successful. At 40 obstacles, MPC–CBF produced a mean path length of 32.009 with a standard deviation of 1.675, and the number of successful runs decreased to 16. Although the increase in mean path length with obstacle density remained comparatively moderate, the decline in success rate in densely cluttered maps indicates increasing difficulty maintaining feasibility as the environment becomes more constrained.

As a result, the MPC–CBF performance indicates a planning strategy that is highly effective and stable in low-to-moderate obstacle densities, where it generates short trajectories with low dispersion and high repeatability. Even as clutter increases, the method preserves comparatively good path quality among successful runs, as reflected by the controlled growth in mean path length and variability. However, its robustness decreases substantially in dense environments, where the growing number of failed runs suggests increasing difficulty in satisfying the optimization and safety constraints simultaneously. This behavior is consistent with the known characteristics of MPC–CBF formulations. While they provide strong safety guarantees and smooth local control in feasible settings, their performance can degrade in highly cluttered maps where narrow corridors and multiple nearby obstacles make feasible constrained solutions harder to obtain.

4.1.7 Particle Swarm Optimization

Standard PSO was examined along with experiments using Gauss–Markov current. The attained results exhibited differentiated behavior when path quality and reliability were considered. In the obstacle-free case, baseline PSO (no current applied) achieved the shortest and most consistent paths, with a mean path length of 27.910 and a standard deviation of 0.075, while all 31 runs were successful. Under the same conditions, the MOD current experiment produced a longer mean path length of 29.292 with a standard deviation of 1.272, also with all runs successful, whereas the STR current variant yielded an even longer mean of 30.368 with a standard deviation of 2.353, but again with 31 successful runs. Thus, without obstacles, the current-affected experiments already showed a loss of path optimality and stability.

For low to moderate clutter levels (2–10 obstacles), baseline PSO maintained 31 successful runs at 2, 6, and 8 obstacles, and only one failure at 4 and 10 obstacles. Its mean path length evolved in a controlled manner from 28.365 at 2 obstacles to 28.962 at 4, 28.653 at 6, 29.755 at 8, and 30.002 at 10 obstacles, with standard deviations ranging from 0.739 to 1.547, indicating comparatively stable performance. The MOD current case had mean path lengths of 29.560, 29.972, 29.598, 30.436, and 29.976 for 2, 4, 6, 8, and 10 obstacles, respectively. However, its robustness began to deteriorate earlier, with success counts of 31, 30, 31, 29, and 27, respectively. The STR current case had higher mean path lengths of 30.172, 30.551, 29.725, 31.023, and 31.266 and success counts of 29, 27, 29, 31, and 28, respectively. Overall, for sparse and moderately cluttered maps, baseline PSO produced the shortest paths with the most consistent success profile, while the MOD and STR current cases led to performance degradation.

The differences became more evident in dense environments. Baseline PSO had 29 successful runs at 20 obstacles, 24 at 30, and 21 at 40. Its corresponding mean path lengths were 31.854, 34.747, and 36.808, showing a gradual deterioration but still remaining comparatively controlled. The MOD current case achieved 23, 18, and 9 successful runs at 20, 30, and 40 obstacles, respectively, with mean path lengths of 32.672, 33.594, and 37.050, respectively. Its trajectory quality among successful runs was sometimes comparable to baseline PSO, but its reliability degraded much more sharply, especially in the most cluttered case. The STR current scenario presented the weakest overall robustness in dense scenarios, with only 22 successes at 20 obstacles, 16 at 30, and 8 at 40. Its mean path lengths were 32.912, 35.311, and 32.722, respectively. Although the mean value at 40 obstacles appears lower than those of the other two PSO variants, this result must be interpreted together with the very low success rate, since it reflects only the subset of runs that successfully reached the goal.

Thus, baseline PSO consistently produces good paths in simple and moderately cluttered environments, but deteriorates in the most clustered scenarios. while also retaining the highest success counts across nearly all obstacle densities. The MOD current case can still produce successful trajectories that are not substantially worse than those of baseline PSO. However, the STR current does substantially impact the performance.

4.2 Statistical Analysis

The Friedman test was applied to the nine examined algorithms of the benchmark. The test revealed statistically significant overall differences among the algorithms, with 188 paired blocks of successful runs examined (the analyses were conducted on the subset of paired runs for which all nine algorithms returned valid path-length values, thus, the statistical ranking reflects comparative path-length performance under complete-case evaluation, rather than robustness to failure, so this complete-case filtering reduced the number of paired blocks from the maximum possible 279 to 188), leading to a Friedman statistic $Q = 502.200$ and $p < 0.001$, confirming that the observed performance variations were not due solely to random fluctuation. According to the mean-rank analysis, CGA achieved the best overall ranking (2.027), followed by MPC–CBF (4.282), PRM–GA (4.660), PSO (4.729), JAYA (4.745), and GWO (4.761), while

APF (5.314), PSO–GM MOD (6.915), and PSO–GM STR (7.569). The pairwise comparisons were further examined using the Nemenyi post hoc test, whose results are presented in Table 3.

Table 3: Complete pairwise comparisons from the Nemenyi post-hoc test. Lower mean rank indicates better overall performance.

Alg1	Alg2	Rank1	Rank2	Diff	Significant
CGA	PSO–GM STR	2.027	7.569	5.543	Yes
CGA	PSO–GM MOD	2.027	6.915	4.888	Yes
APF	CGA	5.314	2.027	3.287	Yes
MPC–CBF	PSO–GM STR	4.282	7.569	3.287	Yes
PRM–GA	PSO–GM STR	4.660	7.569	2.910	Yes
PSO	PSO–GM STR	4.729	7.569	2.840	Yes
JAYA	PSO–GM STR	4.745	7.569	2.824	Yes
GWO	PSO–GM STR	4.761	7.569	2.809	Yes
CGA	GWO	2.027	4.761	2.734	Yes
CGA	JAYA	2.027	4.745	2.718	Yes
CGA	PSO	2.027	4.729	2.702	Yes
CGA	PRM–GA	2.027	4.660	2.633	Yes
MPC–CBF	PSO–GM MOD	4.282	6.915	2.633	Yes
CGA	MPC–CBF	2.027	4.282	2.255	Yes
APF	PSO–GM STR	5.314	7.569	2.255	Yes
PRM–GA	PSO–GM MOD	4.660	6.915	2.255	Yes
PSO	PSO–GM MOD	4.729	6.915	2.186	Yes
JAYA	PSO–GM MOD	4.745	6.915	2.170	Yes
GWO	PSO–GM MOD	4.761	6.915	2.154	Yes
CGA	APF	2.027	5.314	3.287	Yes
APF	PSO–GM MOD	5.314	6.915	1.601	No
APF	GWO	5.314	4.761	0.553	No
APF	JAYA	5.314	4.745	0.569	No
APF	PSO	5.314	4.729	0.585	No
APF	PRM–GA	5.314	4.660	0.654	No
APF	MPC–CBF	5.314	4.282	1.032	No
GWO	JAYA	4.761	4.745	0.016	No
GWO	PSO	4.761	4.729	0.032	No
GWO	PRM–GA	4.761	4.660	0.101	No
GWO	MPC–CBF	4.761	4.282	0.479	No
JAYA	PSO	4.745	4.729	0.016	No
JAYA	PRM–GA	4.745	4.660	0.085	No

(Continued)

Table 3 (continued)

Alg1	Alg2	Rank1	Rank2	Diff	Significant
JAYA	MPC–CBF	4.745	4.282	0.463	No
PSO	PRM–GA	4.729	4.660	0.069	No
PSO	MPC–CBF	4.729	4.282	0.447	No
PRM–GA	MPC–CBF	4.660	4.282	0.378	No
PSO–GM MOD	PSO–GM STR	6.915	7.569	0.654	No

On the one hand, the Friedman and Nemenyi results favored CGA, which achieved the best overall mean rank and statistically outperformed several competing methods in the paired-block comparison. On the other hand, from an application-oriented viewpoint, PRM–GA demonstrated the strongest practical robustness, since it consistently preserved feasibility across all tested obstacle densities, while baseline PSO also proved highly competitive by combining short paths with comparatively strong reliability in many scenarios. This distinction is important because rank-based statistical superiority does not necessarily imply the best operational behavior in dense environments, where success rate and feasibility preservation become essential.

5 Conclusion

This work presented a comparative analysis of representative path-planning approaches for autonomous underwater vehicles, including metaheuristic optimization methods, sampling-based planning, potential-field navigation, and predictive control with safety constraints. The evaluation was conducted in a controlled two-dimensional simulated environment with systematically increasing obstacle density, allowing a consistent assessment of both path quality and feasibility preservation. In addition, the effect of temporally correlated ocean-current disturbances was examined through Gauss–Markov models, with a particular focus on the sensitivity of PSO-based trajectory generation.

The results highlight that algorithm performance is strongly dependent on environmental complexity. In sparse and moderately cluttered scenarios, several methods achieved comparable path quality, with metaheuristic approaches such as PSO, CGA, GWO, and JAYA producing short trajectories with relatively low variability. However, as obstacle density increased, clear differences emerged in robustness. In particular, PRM–GA showed the most consistent behavior, maintaining an ideal success rate across all tested scenarios while exhibiting controlled growth in path length.

The introduction of current disturbances through Gauss–Markov models led to a consistent degradation in PSO performance, with MOD and STR current configurations reducing efficiency and success rate. These results indicate that, within the present formulation, current-induced perturbations increase the difficulty of maintaining stable and feasible trajectories. The statistical analysis further confirmed significant differences among the evaluated algorithms, with CGA achieving the best mean rank in terms of path-length-based performance. However, this ranking does not fully capture practical robustness, as methods such as PRM–GA provided more reliable feasibility preservation under increasing environmental constraints.

Overall, the findings indicate that PRM–GA is the most reliable method for highly cluttered environments due to its explicit encoding of free-space connectivity, while standard optimization methods like PSO can reach an effective balance between efficiency and reliability in moderately complex scenarios. Control-based methods such as MPC–CBF provide high-quality trajectories when feasible, but their success rate

decreases in densely constrained environments, whereas APF and CGA are more sensitive to geometric complexity and exhibit reduced robustness under high clutter.

Despite these contributions, the study presents several limitations. The benchmark is based on a simplified two-dimensional workspace with circular obstacle representations, which, although useful for controlled evaluation, does not fully capture the geometric and physical complexity of real underwater environments. Furthermore, the framework adopts a kinematic abstraction and does not explicitly model vehicle dynamics, sensing uncertainty, or partial observability. As a result, the generated trajectories should be interpreted as geometric planning solutions rather than directly executable paths. In addition, the analysis relies on fixed parameter settings and specific path encodings, which may influence the observed performance.

Future work will focus on extending the framework to more realistic scenarios, including three-dimensional environments, dynamic and uncertain obstacle representations, and the integration of vehicle dynamics and trajectory-tracking control. The incorporation of sensing uncertainty and partial observability is also an important direction to improve realism. Finally, the exploration of hybrid architectures, adaptive parameter tuning, and current-aware cost formulations may provide further improvements in robustness and efficiency under complex environmental conditions.

Acknowledgement: The authors gratefully acknowledge IntellMax–Optimization, Artificial Intelligence and Data Science, Lda. (Portugal) for support during the early phase of this work.

Funding Statement: This work was supported by a PhD Research Scholarship awarded to the first author by ARDITI (Madeira's Regional Agency for the Development of Research, Technology and Innovation) under Operation M2030–FSE+–01277200 (Madeira 2030 Program). This research was also supported by FCT (Portuguese Foundation for Science and Technology) through Projects 10.54499/LA/P/0083/2020 and UID/50009/2025.

Author Contributions: The authors confirm contribution to the paper as follows: Conceptualization & methodology, Fábio Mendonça and Luiz Guerreiro Lopes; software, Rubina Castro and Bruno Silva; validation, Fábio Mendonça, Luiz Guerreiro Lopes and Bruno Silva; investigation & writing—original draft preparation, Rubina Castro; writing—review and editing & supervision, Fábio Mendonça and Luiz Guerreiro Lopes; funding acquisition, Fábio Mendonça. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The data that support the findings of this study are available from the Corresponding Author, Fábio Mendonça, upon reasonable request.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A

Algorithm A1 outlines the proposed MPC–CBF navigation strategy. At each iteration, the agent first checks whether the direct path to the goal is obstacle-free. If the direct path is blocked, a local subgoal is generated around the nearest blocking obstacle. The controller then solves a single MPC–CBF step toward the selected reference. When the optimization problem is infeasible or the resulting motion is invalid, a tangent-based fallback maneuver is applied to preserve progress and obstacle avoidance. The trajectory is iteratively updated until the goal is reached or the maximum number of iterations is exceeded.

Algorithm A1: MPC—CBF path navigation

Input: Start position $\mathbf{x}_s$, goal position $\mathbf{x}_g$, obstacle set $\mathcal{O}$, maximum iterations T, goal threshold G_T, and step size D_T
Output: Trajectory history $\mathcal{H}$, cumulative distances $\mathcal{D}$, fitness history $\mathcal{F}$, and success flag *reached*
Initialize current position $\mathbf{x} \leftarrow \mathbf{x}_s$
Build MPC–CBF solver for obstacle set $\mathcal{O}$
Initialize history $\mathcal{H} \leftarrow [\mathbf{x}]$
Initialize trajectory list $\mathcal{P} \leftarrow [\mathbf{x}]$
Initialize fitness list $\mathcal{F} \leftarrow [\,]$
Initialize cumulative-distance list $\mathcal{D} \leftarrow [\,]$
Initialize total distance $D_{\text{tot}} \leftarrow 0$
Set *reached* ← **false**
for $k = 1$ to T **do**
    if the segment from $\mathbf{x}$ to $\mathbf{x}_g$ intersects any obstacle **then**
        Generate local reference $\mathbf{x}_{ref}$ around the nearest blocking obstacle
    else
        Set $\mathbf{x}_{ref} \leftarrow \mathbf{x}_g$
    end if
    Recover previous MPC solution if available
    Warm-start the MPC problem
    Update the optimization problem with current state $\mathbf{x}$ and reference $\mathbf{x}_{ref}$
    Reconstruct the linearized CBF constraints for all obstacles
    Solve the MPC problem
    if the optimization is infeasible or no valid control is obtained **then**
        Compute a tangent-based fallback motion
        Set candidate position $\mathbf{x}_{cand}$ from the fallback rule
    else
        Extract the first control action $\mathbf{u}_1$
        Compute candidate position: $\mathbf{x}_{cand} \leftarrow \mathbf{x} + D_T \cdot \mathbf{u}_1$
        Clamp $\mathbf{x}_{cand}$ to workspace bounds
        if $\mathbf{x}_{cand}$ is invalid, too close to $\mathbf{x}$, or collides with an obstacle **then**
            Compute a tangent-based fallback motion
            Update $\mathbf{x}_{cand}$ accordingly
        end if
    end if
    Update current position: $\mathbf{x} \leftarrow \mathbf{x}_{cand}$
    Append $\mathbf{x}$ to $\mathcal{H}$
    if $|\mathcal{H}| > 20$ **then**
        Remove the oldest element from $\mathcal{H}$
    end if
    Append $\mathbf{x}$ to $\mathcal{P}$
    Update traveled distance: $D_{\text{tot}} \leftarrow D_{\text{tot}} + \|\mathcal{P}_{end} - \mathcal{P}_{end-1}\|$
    Append D_{tot} to $\mathcal{D}$

(Continued)

Algorithm A1 (continued)

Evaluate fitness at $\mathbf{x}$ and append it to $\mathcal{F}$
if $\|\mathbf{x} - \mathbf{x}_g\| < G_T$ **then**
  Set $\mathbf{x} \leftarrow \mathbf{x}_g$
  Append $\mathbf{x}$ to $\mathcal{P}$
  Update traveled distance: $D_{\text{tot}} \leftarrow D_{\text{tot}} + \|\mathcal{P}_{end} - \mathcal{P}_{end-1}\|$
  Append D_{tot} to $\mathcal{D}$
  Evaluate final fitness and append it to $\mathcal{F}$
  Set *reached* ← **true**
  break
end if
end for
Pad $\mathcal{F}$ and $\mathcal{D}$ to length T
if *reached* = **false then**
  Replace cumulative distances with failure markers
end if
return $\mathcal{H}, \mathcal{D}, \mathcal{F}$, *reached*

Algorithm A2 outlines the proposed PRM–GA navigation strategy. First, a PRM is constructed in the free space by sampling collision-free nodes and connecting neighboring nodes through valid local edges. To accelerate neighborhood queries, a KD-tree is built over the sampled nodes and used to retrieve the nearest candidates for connection. Then, a GA evolves waypoint sequences over the roadmap graph in order to obtain a feasible path from start to goal. If the initial roadmap fails to provide a valid goal-reaching solution, a denser roadmap is generated as a fallback. The resulting path is subsequently refined through shortcutting, arc-based repair, and smoothing, followed by a final collision check.

Algorithm A2: PRM–GA path navigation

Input: Start position $\mathbf{x}_s$, goal position $\mathbf{x}_g$, obstacle set $\mathcal{O}$, roadmap sample count S, neighborhood size k, benchmark length T, and goal threshold G_T
Output: Trajectory history $\mathcal{H}$, cumulative distances $\mathcal{D}$, fitness history $\mathcal{F}$, and success flag *reached*
Initialize fitness list $\mathcal{F} \leftarrow [\,]$
Initialize trajectory list $\mathcal{P} \leftarrow [\,]$
Initialize cumulative-distance list $\mathcal{D} \leftarrow [\,]$
Initialize total distance $D_{\text{tot}} \leftarrow 0$
Set *reached* ← **false**
Generate S collision-free roadmap samples, including $\mathbf{x}_s$ and $\mathbf{x}_g$
Build a KD-tree over the sampled roadmap nodes
for each roadmap node **do**
  Retrieve its nearest neighbors using KD-tree search
  Validate each candidate edge through collision checking
  Add collision-free connections to the roadmap graph
end for
Run the genetic algorithm over roadmap-node indices to obtain a candidate path

(Continued)

Algorithm A2 (continued)

Attempt to reconnect the candidate path to the goal using shortest-path search on the roadmap
if no valid goal-reaching path is found **then**
    Rebuild a denser roadmap with additional samples
    Reconstruct the KD-tree and roadmap graph
    Run the genetic algorithm again
    Attempt again to reconnect the candidate path to the goal
    if no valid path is found **then**
        Return failure result with invalid distance markers
    end if
end if
if the obtained path is empty or does not terminate at the goal **then**
    Return failure result
end if
Apply shortcutting to reduce unnecessary detours
Repair colliding segments using arc-based local replacements
Smooth the repaired path
Subdivide the path into finer trajectory points
if the smoothed path collides with obstacles **then**
    Revert to the repaired unsmoothed version
end if
if the path still collides **then**
    Rebuild an even denser roadmap
    Reconstruct the KD-tree and roadmap graph
    Re-run the genetic search and goal reconnection
    if no valid goal-reaching path is found **then**
        Return failure result
    end if
    Repair, smooth, and subdivide the new path
    if the smoothed version still collides **then**
        Revert to the repaired unsmoothed version
    end if
end if
if the final path does not terminate at the goal **then**
    if its last point can be connected directly to the goal without collision **then**
        Append the goal point to the path
    else
        Return failure result
    end if
end if
if the final path collides or does not satisfy G_T **then**
    Return failure result
end if
Convert the final path into trajectory history $\mathcal{H}$

(Continued)

Algorithm A2 (continued)

59: **if** $|\mathcal{H}| < 2$ **then**
60: Duplicate the final point to ensure a minimum trajectory length
61: **end if**
62: **for** each point $\mathbf{p}_i$ in $\mathcal{H}$ **do**
63: Evaluate fitness using the partial trajectory history
64: Append the result to $\mathcal{F}$
65: **end for**
66: Initialize $\mathcal{D} \leftarrow [0]$
67: **for** each consecutive pair $(\mathbf{p}_i, \mathbf{p}_{i+1})$ in $\mathcal{H}$ **do**
68: $D_{\text{tot}} \leftarrow D_{\text{tot}} + \|\mathbf{p}_{i+1} - \mathbf{p}_i\|$
69: Append D_{tot} to $\mathcal{D}$
70: **end for**
71: Set $\mathcal{P} \leftarrow \mathcal{H}$
72: Pad $\mathcal{P}$, $\mathcal{D}$, and $\mathcal{F}$ to length T
73: Set *reached* $\leftarrow$ **true**
74: **return** $\mathcal{H}, \mathcal{D}, \mathcal{F}$, *reached*

References

1. Xu X, Zeng J, Zhao Y, Lü X. Research on global path planning algorithm for mobile robots based on improved A*. Expert Syst Appl. 2024;243(7):122922. doi:10.1016/j.eswa.2023.122922.
2. Ullah I, Adhikari D, Khan H, Anwar MS, Ahmad S, Bai X. Mobile robot localization: current challenges and future prospective. Comput Sci Rev. 2024;53:100651. doi:10.1016/j.cosrev.2024.100651.
3. LaValle SM. Planning algorithms. New York, NY, USA: Cambridge University Press; 2006.
4. Wynn RB, Huvenne VAI, Le Bas TP, Murton BJ, Connelly DP, Bett BJ, et al. Autonomous underwater vehicles (AUVs): their past, present and future contributions to the advancement of marine geoscience. Mar Geol. 2014;352:451–68. doi:10.1016/j.margeo.2014.03.012.
5. Paull L, Saeedi S, Seto M, Li H. AUV navigation and localization: a review. IEEE J Ocean Eng. 2014;39(1):131–49. doi:10.1109/joe.2013.2278891.
6. Kot R. Review of collision avoidance and path planning algorithms used in autonomous underwater vehicles. Electronics. 2022;11(15):2301. doi:10.3390/electronics11152301.
7. Panda M, Das B, Subudhi B, Pati BB. A comprehensive review of path planning algorithms for autonomous underwater vehicles. Int J Autom Comput. 2020;17(3):321–52. doi:10.1007/s11633-019-1204-9.
8. Cheng C, Sha Q, He B, Li G. Path planning and obstacle avoidance for AUV: a review. Ocean Eng. 2021;235(7):109355. doi:10.1016/j.oceaneng.2021.109355.
9. An D, Mu Y, Wang Y, Li B, Wei Y. Intelligent path planning technologies of underwater vehicles: a review. J Intell Robot Syst. 2023;107(2):22. doi:10.1007/s10846-022-01794-y.
10. Kavraki LE, Svestka P, Latombe JC, Overmars MH. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Trans Robot Autom. 1996;12(4):566–80. doi:10.1109/70.508439.
11. Kannan A, Gupta P, Tiwari R, Prasad S, Khatri A, Kala R. Robot motion planning using adaptive hybrid sampling in probabilistic roadmaps. Electronics. 2016;5(2):16. doi:10.3390/electronics5020016.
12. Alvarez A, Caiti A, Onken R. Evolutionary path planning for autonomous underwater vehicles in a variable ocean. IEEE J Ocean Eng. 2004;29(2):418–29. doi:10.1109/joe.2004.827837.
13. Yan S, Pan F. Research on route planning of AUV based on genetic algorithms. In: 2019 IEEE International Conference on Unmanned Systems and Artificial Intelligence (ICUSAI). Piscataway, NJ, USA: IEEE; 2019. p. 184–7.
14. Haupt RL, Haupt SE. Practical genetic algorithms. 2nd ed. Hoboken, NJ, USA: Wiley; 2004.

15. Kennedy J, Eberhart R. Particle swarm optimization. In: Proceedings of ICNN'95—International Conference on Neural Networks. Piscataway, NJ, USA: IEEE; 1995. p. 1942–8.
16. Freitas D, Lopes LG, Morgado-Dias F. Particle swarm optimisation: a historical review up to the current developments. Entropy. 2020;22(3):362. doi:10.3390/e22030362.
17. Mirjalili S, Mirjalili SM, Lewis A. Grey wolf optimizer. Adv Eng Softw. 2014;69:46–61. doi:10.1016/j.advengsoft.2013.12.007.
18. Li J, Xu H, Tian Z. Global path planning for AUV based on improved grey wolf optimization. In: 2024 IEEE International Conference on Mechatronics and Automation (ICMA). Piscataway, NJ, USA: IEEE; 2024. p. 1379–84.
19. Rao RV. Jaya: a simple and new optimization algorithm for solving constrained and unconstrained optimization problems. Int J Ind Eng Comput. 2016;7:19–34.
20. Silva B, Lopes LG, Mendonça F. Parallel GPU-acceleration of metaphorless optimization algorithms: application for solving large-scale nonlinear equation systems. Appl Sci. 2024;14(12):5349.
21. Fan X, Guo Y, Liu H, Wei B, Lyu W. Improved artificial potential field method applied for AUV path planning. Math Probl Eng. 2020;2020:6523158. doi:10.1155/2020/6523158.
22. Yan Z, Zhao L, Wang Y, Zhang M, Yang H, Zhang C. Path planning of AUV for obstacle avoidance with improved artificial potential field. In: IECON 2023—49th Annual Conference of the IEEE Industrial Electronics Society. Piscataway, NJ, USA: IEEE; 2023. p. 1–5.
23. Tian Z, Chen M. Safe and optimal motion planning for autonomous underwater vehicles: a robust model predictive control framework integrating fast marching time objectives and adaptive control barrier functions. Drones. 2025;9(4):273. doi:10.3390/drones9040273.
24. Huang J, Wang H, Margellos K, Goulart P. Predictive control barrier functions: bridging model predictive control and control barrier functions. In: 2025 European Control Conference (ECC). Piscataway, NJ, USA: IEEE; 2025. p. 2623–9.
25. Garau B, Alvarez A, Oliver G. Path planning of autonomous underwater vehicles in current fields with complex spatialvariability: an A* approach. In: Proceedings of the 2005 IEEE International Conference on Robotics and Automation. Piscataway, NJ, USA: IEEE; 2005. p. 194–8.
26. Arifi A, Bouallègue S, Lepagnot J, Jourdan L. Probabilistic roadmap-based 3D path planning of autonomous underwater vehicles. In: 2023 IEEE Third International Conference on Signal, Control and Communication (SCC). Piscataway, NJ, USA: IEEE; 2023. p. 1–6.
27. Zhao E, Cheng T, Wang S, Lin X. Underwater 3D path planning for AUV in ocean currents based on improved informed RRT* algorithm. J Intell Robot Syst. 2025;111(4):116. doi:10.1007/s10846-025-02320-6.
28. Li H, Li S, Wang Q, Huang X. AUV 3D path planning based on improved PSO. J Syst Eng Electron. 2025;36(3):854–66. doi:10.23919/jsee.2025.000074.
29. Meng R, Sun A, Wu Z, Du X, Meng Y. 3D smooth path planning of AUV based on improved ant colony optimization considering heading switching pressure. Sci Rep. 2023;13(1):14076. doi:10.1038/s41598-023-41140-2.
30. Vu DC, Tran S, Nguyen TL, Hoang DC. Glocal trajectory generation and tracking control for autonomous underwater vehicles with optimal coverage sensor networks. Ocean Eng. 2025;342(8):122902. doi:10.1016/j.oceaneng.2025.122902.
31. Garau B, Bonet M, Alvarez A, Ruiz S, Pascual A. Path planning for autonomous underwater vehicles in realistic oceanic current fields: application to gliders in the Western Mediterranean Sea. J Marit Res. 2009;6(2):5–21.
32. Zeng Z, Sammut K, Lian L, He F, Lammas A, Tang Y. A comparison of optimization techniques for AUV path planning in environments with ocean currents. Robot Auton Syst. 2016;82(1):61–72. doi:10.1016/j.robot.2016.03.011.
33. Khatib O. Real-time obstacle avoidance for manipulators and mobile robots. Int J Robot Res. 1986;5(1):90–8. doi:10.1177/027836498600500106.
34. Koren Y, Borenstein J. Potential field methods and their inherent limitations for mobile robot navigation. In: Proceedings of the 1991 IEEE International Conference on Robotics and Automation. Piscataway, NJ, USA: IEEE; 1991. p. 1398–404.

35. Solari FJ, Rozenfeld AF, Villar SA, Acosta GG. Artificial potential fields for the obstacles avoidance system of an AUV using a mechanical scanning sonar. In: 2016 3rd IEEE/OES South American International Symposium on Oceanic Engineering (SAISOE). Piscataway, NJ, USA: IEEE; 2016. p. 1–6.
36. Fu J, Lv T, Li B. Underwater submarine path planning based on artificial potential field ant colony algorithm and velocity obstacle method. Sensors. 2022;22(10):3652. doi:10.3390/s22103652.
37. Tang S, Tong C, Chen X, Yue J, Wang H. Research on an autonomous underwater vehicle intelligent path planning algorithm based on improved artificial potential field method optimization. In: Proceedings of the 2024 8th International Conference on Electronic Information Technology and Computer Engineering. New York, NY, USA: ACM; 2024. p. 1241–7.
38. Deb K, Pratap A, Agarwal S, Meyarivan T. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Comput. 2002;6(2):182–97. doi:10.1109/4235.996017.
39. Toma AI, Hsueh HY, Jaafar HA, Murai R, Kelly PHJ, Saeedi S. PathBench: a benchmarking platform for classical and learned path planning algorithms. In: 2021 18th Conference on Robots and Vision (CRV). Piscataway, NJ, USA: IEEE; 2021. p. 79–86.
40. Yang D, Xiong Y. UP-Bench: a benchmark for underwater path planning algorithms. In: Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'25). New York, NY, USA: ACM; 2025. p. 5854–65.
41. Mayne DQ, Rawlings JB, Rao CV, Scokaert POM. Constrained model predictive control: stability and optimality. Automatica. 2000;36(6):789–814.
42. Ames AD, Xu X, Grizzle JW, Tabuada P. Control barrier function based quadratic programs for safety critical systems. IEEE Trans Autom Control. 2017;62(8):3861–76. doi:10.1109/tac.2016.2638961.
43. Shankar M, Gangireddy S. A hybrid path planning approach combining artificial potential field and particle swarm optimization for mobile robot. IFAC-PapersOnLine. 2022;55(22):242–7. doi:10.1016/j.ifacol.2023.03.041.
44. Zheng L, Yu W, Li G, Qin G, Luo Y. Particle swarm algorithm path-planning method for mobile robots based on artificial potential fields. Sensors. 2023;23(13):6082. doi:10.3390/s23136082.
45. Zhao Z, Zhang Y, Feng X, Jiang C, Su W, Hu Q. A dynamic velocity potential field method for multi-AUV cooperative hunting tasks. Ocean Eng. 2024;295(3):116813. doi:10.1016/j.oceaneng.2024.116813.
46. Sun B, Niu N. Multi-AUVs cooperative path planning in 3D underwater terrain and vortex environments based on improved multi-objective particle swarm optimization algorithm. Ocean Eng. 2024;311(6):118944. doi:10.1016/j.oceaneng.2024.118944.
47. Wabersich KP, Taylor AJ, Choi JJ, Sreenath K, Tomlin CJ, Ames AD, et al. Data-driven safety filters: hamilton-Jacobi reachability, control barrier functions, and predictive methods for uncertain systems. IEEE Control Syst Mag. 2023;43(5):137–77. doi:10.1109/MCS.2023.3291885.
48. Keyumarsi S, Atman MWS, Gusrialdi A. Circulation-embedded control barrier function for safe navigation: a solution to avoid undesired equilibria and dysfunctional circulation. Robot Auton Syst. 2025;194:105132. doi:10.1016/j.robot.2025.105132.
49. Aldhaheri S, Hu Y, Xie Y, Wu P, Kanoulas D, Liu Y. Underwater robotic simulators review for autonomous system development. In: OCEANS 2025 Brest. Piscataway, NJ, USA: IEEE; 2025. p. 1–10.
50. Cieślak P. Stonefish: an advanced open-source simulation tool designed for marine robotics, with a ROS interface. In: OCEANS 2019 Marseille. Piscataway, NJ, USA: IEEE; 2019. p. 1–6.
51. Marchesini E, Corsi D, Farinelli A. Benchmarking safe deep reinforcement learning in aquatic navigation. In: 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Piscataway, NJ, USA: IEEE; 2021. p. 5590–5.
52. Liu Y, Zhang J, Xiang X, Liu J. Design, implementation and verification of hardware-in-the-loop control system for work-class ROVs. Ocean Eng. 2024;313(2):119605. doi:10.1016/j.oceaneng.2024.119605.
53. Yin S, Xiang Z. ADE-MIFS: real-time cooperative navigation for multi-USVs in dynamic maritime environments. Ocean Eng. 2026;352:124487. doi:10.1016/j.oceaneng.2026.124487.
54. Rezaei H, Bozorg-Haddad O, Chu X. Grey wolf optimization (GWO) algorithm. In: Bozorg-Haddad O, editor. Advanced optimization by nature-inspired algorithms. Singapore: Springer; 2018. p. 81–91.

55. Ferguson D, Likhachev M, Stentz A. A guide to heuristic-based path planning. In: Proceedings of ICAPS'05 Workshop on Planning under Uncertainty for Autonomous Systems. Menlo Park, CA, USA: AAAI Press; 2005. p. 9–18.
56. Martínez A, Hernández L, Sahli H, Valeriano Y, Orozco Monteagudo M, Garcia D. Model-aided navigation with sea current estimation for an autonomous underwater vehicle. Int J Adv Robot Syst. 2015;12(7):103. doi:10.5772/60415.
57. Chen X, Bian H, Li F, Wang R, Hu Y, Li J. Time-varying current estimation method for SINS/DVL integrated navigation based on augmented observation algorithm. Symmetry. 2025;17(11):1881. doi:10.3390/sym17111881.