foundations

MDPI

Article

A Multiscale Hybrid Algorithm for Solving Zermelo's Problem Using Local Analytical Solutions and A-Based Graph Search

José Antonio González-Prieto

Defense University Center, Spanish Naval Academy, Plaza de España, s/n, 36920 Marín, Spain; jose.gonzalez@cud.uvigo.es

Abstract

This paper presents a hybrid search algorithm for Zermelo's navigation problem in complex vector fields. The method combines local analytical navigation with A*-based graph search to compute high-quality trajectories in scenarios with multiple local optima. Its key idea is to use the optimal navigation law for uniform vector fields as a local propagation rule to generate new nodes. By defining multiple target points and linking them through locally optimal analytical trajectories, the algorithm encourages structured global exploration beyond purely local methods. The resulting framework integrates analytical local navigation with the robustness of heuristic search to integrate global exploration properties. It also adapts automatically to different spatial scales, making it suitable for both local and large-domain problems. Numerical validation under execution-time constraints demonstrates its effectiveness in time-limited settings, with experiments showing strong global exploration and high-quality trajectories under strict computational budgets.

Keywords: Zermelo's navigation problem; time optimal control; hybrid path planning; A* algorithm

1. Introduction

Path planning of autonomous marine vehicles under environmental perturbations remains a challenging problem, especially when the objective is to compute trajectories that are feasible, safe, and near-optimal under realistic flow conditions. A classical model for this setting is the Zermelo navigation problem, originally formulated for minimum-time navigation in a prescribed flow field. For steady and uniform current fields, the continuous Zermelo navigation problem with direct heading control admits an analytical optimal solution, commonly characterized through the velocity-triangle construction and optimal-control arguments. However, such closed-form relations become insufficient when the current field varies in space or time, when obstacles and local flow distortions alter the feasible navigation domain, or when additional modeling constraints are introduced, such as bounded heading-rate control or a discrete path-planning formulation [1–5].

In practical marine environments, currents are rarely uniform. Spatially varying flows, vortices, opposing currents, moving obstacles, wind, and waves can induce nonlinear interactions between the vehicle dynamics and the environment. Consequently, several qualitatively different trajectories may coexist, and algorithms based only on local optimality criteria may converge to suboptimal routes. This is particularly relevant for autonomous underwater vehicles (AUVs), unmanned surface vehicles (USVs), and autonomous ships, where a planner must balance global exploration, local dynamic feasibility, safety, and computational efficiency.

Check for updates

Academic Editor: Hsien-Chung Wu

Received: 30 July 2026
Revised: 26 August 2026
Accepted: 29 September 2026
Published: 3 October 2026

Copyright: © 2026 by the author. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

A broad range of methods has been developed to address these difficulties, such as graph-based planners that extend classical search strategies by incorporating hydrodynamic information, spatio-temporal feasibility constraints, or flow-dependent edge costs. They provide systematic global exploration, but their performance can depend strongly on grid resolution, current-field accuracy, and local transition models. Representative examples include route-search graphs from ship-trajectory data, graph search coupled with differential equations, constrained A^* formulations, A^* combined with artificial potential fields (APF), current-dependent global planners and streamline or computational fluid dynamics-based approaches [6–11].

Probabilistic roadmap (PRM) methods improve flexibility in cluttered or partially structured regions, especially when sampling is biased toward navigable or hydrodynamically favorable areas. Random-sampling approaches can also generate feasible paths in obstacle-rich environments, although their cost may increase in large-scale or densely constrained maps. Collision-avoidance methods focus on safe maneuvers and compliance with the International Regulations for Preventing Collisions at Sea (COLREG), but they are usually designed for encounter resolution rather than global optimality [12–15].

Optimization and metaheuristic strategies have also been widely investigated for marine path planning in nonlinear and uncertain environments. Particle swarm optimization (PSO), visibility graphs, the Whale Optimization Algorithm (WOA), Bayesian current forecasting, Gaussian-process motion-planning formulations, and comparative optimization frameworks have been used to compute energy-efficient, smooth, or dynamically feasible trajectories. Other works address dynamic current fields, hydrodynamic ship tracking, and time-optimal maneuvering in close-range encounters. These methods can capture complex objectives and constraints, but often require careful parameterization and substantial computational effort [16–22].

Learning-based and adaptive methods have been proposed to improve robustness in uncertain or time-varying environments. Reinforcement learning can learn navigation policies without relying exclusively on explicit environmental models, while fuzzy neural networks and related schemes improve local decision-making under nonlinear disturbances. Nevertheless, these approaches may require extensive training, can be difficult to interpret, and do not always provide global optimality guarantees. Several studies also consider the combined influence of wind, waves, and currents, as well as energy–time trade-offs for marine vehicles under dynamic disturbances [23–28].

Recent hybrid approaches seek to combine global search, local dynamic fidelity, and adaptive decision-making. Hybrid search methods for Zermelo-type problems aim to balance tractability with near-optimality; APF and globally guided reinforcement learning have been combined for USV navigation in dynamic environments; improved A^* search has been fused with the dynamic window approach (DWA) for smoother obstacle avoidance; and synergistic differential evolution has been used for energy-efficient AUV path planning under multiple constraints. Review studies further indicate that reproducibility and fair comparison remain limited because simulation conditions, environmental models, and code availability vary substantially across the literature [29–33].

Despite this diversity, an important methodological gap remains. Many existing approaches emphasize either global search structure or local dynamic fidelity, but not both simultaneously. Graph-based methods explore large regions systematically, yet their transitions are often based on geometric adjacency or discretized motion primitives that do not fully exploit the analytical structure of the navigation problem. Conversely, continuous optimal-control formulations and local steering laws represent the dynamics more faithfully, but they may struggle in multimodal flow fields where several competing trajectory families

exist. This tension becomes critical when a planner must operate across scales, from local navigation to wide-area routing, while maintaining reasonable computational cost.

This paper addresses this issue through a hybrid search technique for the Zermelo navigation problem in general vector fields. The proposed approach combines local analytical navigation with an A^*-based graph-search framework. The central idea is to use the optimal-control law associated with uniform-current navigation as a local propagation mechanism for constructing dynamically meaningful graph structures. Instead of relying only on traditional geometric neighborhoods, the search nodes are generated through locally guided short trajectories, thereby combining dynamic feasibility with efficient global exploration.

The resulting framework is intended for scenarios in which multiple locally optimal trajectory families may exist and where the characteristic scale of the planning problem can vary significantly. The method is not proposed as a benchmark of existing techniques, but as a navigation strategy that connects analytical local guidance with global heuristic exploration. Its goal is to generate high-quality paths in complex flow environments while preserving computational efficiency under prescribed time limitations.

2. Zermelo Navigation Problem

2.1. Zermelo Navigation Problem Formulation

The Zermelo navigation problem is a classical formulation in nonlinear optimal control that seeks the time-optimal trajectory of a vessel navigating within a known current field; see [1]. The state is defined as

$$\mathbf{z}(t) = \begin{bmatrix} x(t) \\ y(t) \\ \psi(t) \end{bmatrix} \in \mathbb{R}^3 \tag{1}$$

where $x(t)$ and $y(t)$ denote the vessel's planar position and $\psi(t)$ its heading angle. The vessel moves with constant surge speed relative to the surrounding fluid, $v_s > 0$, which is treated as a fixed model parameter rather than a control variable. This produces the body-fixed velocity vector

$$\mathbf{v}_s(t) = v_s \begin{bmatrix} \cos\psi(t) \\ \sin\psi(t) \end{bmatrix} \tag{2}$$

The ocean current is modeled as a known, potentially spatially varying velocity field

$$\mathbf{v}_c(x, y) = \begin{bmatrix} v_{cx}(x, y) \\ v_{cy}(x, y) \end{bmatrix} \tag{3}$$

which acts as an external disturbance on the vehicle motion. If the heading angle is directly selected as the control variable, i.e., $\psi(t) = u(t)$, while the surge speed v_s remains constant, the ground-referenced dynamics are

$$\begin{aligned} \dot{x}(t) &= v_s \cos\psi(t) + v_{cx}(x, y) \\ \dot{y}(t) &= v_s \sin\psi(t) + v_{cy}(x, y) \end{aligned} \tag{4}$$

The corresponding time-optimal control problem consists of finding a heading profile $\psi(t)$ and a final time T, for fixed surge speed v_s, such that the vessel moves from an initial state $\mathbf{z}(0) = \mathbf{z}_{\text{start}}$ to a desired point $\mathbf{z}_{\text{goal}}$ while minimizing the total travel time:

$$\begin{aligned} \min_{\psi(\cdot),\,T} \quad & T \\ \text{subject to} \quad & (4) \\ & \mathbf{z}(0) = \mathbf{z}_{\text{start}} \\ & \mathbf{z}(T) = \mathbf{z}_{\text{goal}} \end{aligned} \tag{5}$$

This optimal control problem is nonlinear due to the trigonometric dependence on $\psi(t)$ and the spatial variation of $\mathbf{v}_c(x, y)$. In nontrivial current fields, multiple locally optimal solutions may exist, some of which may sacrifice distance for time or vice versa. This makes the problem challenging not only analytically but also computationally, since global optimality is not guaranteed by local solvers.

2.1.1. Solutions in Uniform Current Fields

When the current field is uniform and constant, i.e.,

$$\mathbf{v}_c(x, y) = \begin{bmatrix} v_{cx} \\ v_{cy} \end{bmatrix}$$

the problem becomes significantly more tractable and admits a closed-form analytical solution. Let the current have direction β_c and magnitude $v_c = |\mathbf{v}_c|$. At any given time, the line-of-sight angle from the vessel position to the target is

$$\beta = \text{atan2}\Big(y_{\text{goal}} - y(t),\, x_{\text{goal}} - x(t)\Big)$$

For minimum-time navigation, the ground velocity must be directed toward the target, i.e., collinear with the displacement vector from the vessel to the goal. This condition defines the classical velocity triangle relating the vessel velocity $\mathbf{v}_s$, the current $\mathbf{v}_c$, and the resulting ground velocity. By the law of sines,

$$\frac{v_s}{\sin(\beta - \beta_c)} = \frac{v_c}{\sin(\psi - \beta)} \tag{6}$$

Solving for the optimal heading gives the analytical control law

$$\psi^* = \beta - \arcsin\left(\frac{v_c}{v_s}\sin(\beta - \beta_c)\right) \tag{7}$$

Equation (7) provides the heading that ensures minimum travel time in a constant current field under a continuous dynamic model. Since it is available in closed form, it is commonly used as a benchmark to validate numerical solvers in simplified scenarios before considering more realistic flow conditions.

The closed-form expression in (7) is defined only when the argument of the inverse sine lies within the interval $[-1, 1]$. Therefore, the velocity-triangle construction requires

$$\left|\frac{v_c}{v_s}\sin(\beta - \beta_c)\right| \leq 1 \tag{8}$$

This inequality is a geometric feasibility requirement of the velocity-triangle construction, rather than a restriction imposed on the heading itself. It expresses the requirement that the component of the current perpendicular to the line of sight, $|v_c \sin(\beta - \beta_c)|$, must not exceed the vessel speed v_s. If this requirement is violated, the vessel velocity cannot compensate for the transverse component of the current, and no heading satisfying the corresponding velocity-triangle construction exists for that vessel–target configuration. This situation can occur when $v_c > v_s$ for unfavorable orientations between the current

and the line of sight. However, $v_c > v_s$ does not by itself imply violation of the inequality, since feasibility also depends on the relative angle $\beta - \beta_c$. Satisfaction of the inequality means that the inverse-sine term is real and that cancellation of the transverse current component is geometrically possible; it should not be interpreted as removing the feasibility requirement when the current is strong.

This geometric requirement is also distinct from the heading-rate constraint introduced in the subsequent formulation. The analytical law above assumes that the heading can be selected directly, whereas the practical model uses the heading rate as the control input and limits it through a maximum admissible turning rate. Consequently, a heading provided by the analytical law may be geometrically defined but still be incompatible with the bounded-turning-rate dynamics. In the proposed graph-search algorithm, when the analytical propagation toward a selected intermediate target is not defined because the inverse-sine requirement is violated, that propagation is not used for the corresponding expansion. The search can instead continue through alternative intermediate targets and dynamically feasible motion segments. The same principle applies when an analytically defined propagation is incompatible with the imposed motion constraints.

2.1.2. Solutions in Non-Uniform Current Fields

In non-uniform and time-varying current fields, closed-form navigation laws are generally insufficient, so most approaches rely on numerical or data-driven planning. Existing methods include current-aware graph search, constrained A^*, streamline-based planning, PRM and random-sampling roadmaps, collision-avoidance schemes, metaheuristic optimization, and learning-based control [6–28].

Although these methods improve feasibility, robustness, or energy efficiency in complex marine environments, they also present recurring limitations: sensitivity to discretization and current-field accuracy, dependence on sampling or heuristic quality, high computational cost, training burden, or lack of global optimality guarantees. Hybrid strategies have therefore emerged to combine global exploration with local dynamic guidance, but fair comparison remains difficult because benchmarks, environmental models, and code availability are not yet standardized [29,33].

2.2. Continuous Model with Soft Termination Condition and Heading Rate as Control Variable

In numerical implementations, enforcing the exact terminal constraint $\mathbf{z}(T) = \mathbf{z}_{\text{goal}}$ may be unnecessarily restrictive and numerically challenging, especially after discretization or when the final heading is not prescribed. A practical alternative is to define arrival through a soft terminal condition: the vessel is considered to have reached the goal when its final position lies within a prescribed tolerance distance from the target. Denoting the planar position by $p(t)$ as

$$p(t) = \begin{bmatrix} x(t) \\ y(t) \end{bmatrix}, \qquad p_{\text{goal}} = \begin{bmatrix} x_{\text{goal}} \\ y_{\text{goal}} \end{bmatrix},$$

the terminal condition is written as

$$\left\| p(T) - p_{\text{goal}} \right\|_2 \leq \varepsilon, \qquad \varepsilon > 0. \tag{9}$$

This soft termination condition is adopted as part of the continuous formulation before discretization, so the discrete optimization problem can be stated directly with the same arrival criterion.

In the previous formulation, the heading angle $\psi(t)$ was directly chosen as the control variable. However, in many practical applications this is not realistic, since vessels are

physically limited by their maximum turning rate. Moreover, the vessel speed is assumed constant in the Zermelo model and is not optimized. A more natural formulation is therefore obtained by considering the heading rate

$$r(t) = \dot{\psi}(t)$$

as the only control input. The surge speed v_s remains fixed, but the dynamics become

$$\begin{aligned} \dot{x}(t) &= v_s \cos\psi(t) + v_{cx}(x,y) \\ \dot{y}(t) &= v_s \sin\psi(t) + v_{cy}(x,y) \\ \dot{\psi}(t) &= r(t) \end{aligned} \tag{10}$$

The resulting time-optimal control problem is formulated as

$$\begin{aligned} \min_{r(\cdot),T} \quad & T \\ \text{subject to} \quad & (10), \\ & \mathbf{z}(0) = \mathbf{z}_{\text{start}} \\ & \left\| p(T) - p_{\text{goal}} \right\|_2 \leq \varepsilon \\ & |r(t)| \leq r_{\max} \end{aligned} \tag{11}$$

where $r_{\max}$ represents the vessel's maximum admissible turning rate. This formulation offers two key advantages. First, by explicitly modeling the heading rate as the control variable while keeping the surge speed constant, the resulting trajectories are inherently dynamically feasible, avoiding the need for post-processing or trajectory smoothing. Second, physical steering constraints can be directly enforced through bounds on $u(t)$, ensuring that solutions remain practically realizable.

The introduction of control bounds modifies the structure of the optimal solution, therefore in uniform current fields, the classical analytical heading law may no longer be optimal or feasible under turning-rate limits. In such cases, alternative trajectories satisfying the steering constraints may provide the true time-optimal solution, highlighting the trade-off between analytical optimality and physical realizability in marine navigation.

2.3. Discrete Model with Soft Termination Condition and Heading Rate as Control Variable

To solve the Zermelo navigation problem numerically, the continuous-time dynamics with constant surge speed and heading rate as the only control input are reformulated as a discrete-time system. This allows the trajectory to be represented as a finite sequence of states and controls while preserving the soft terminal condition introduced in the continuous model.

2.3.1. Discrete-Time Model via Euler Discretization

Let the time horizon be discretized into N intervals of fixed step size $\Delta t > 0$, such that

$$t_k = k\Delta t, \qquad k = 0, 1, \ldots, N, \qquad T = N\Delta t.$$

The vessel state at step k is

$$\mathbf{z}_k = \begin{bmatrix} x_k \\ y_k \\ \psi_k \end{bmatrix},$$

and the control input is the heading rate, while v_s remains constant,

$$r_k = \frac{\psi_{k+1} - \psi_k}{\Delta t}, \qquad |r_k| \le r_{\max}.$$

Using the forward Euler method, the dynamics are approximated by

$$x_{k+1} = x_k + \Delta t\Big(v_s \cos\psi_k + v_{cx}(x_k, y_k)\Big), \tag{12}$$

$$y_{k+1} = y_k + \Delta t\Big(v_s \sin\psi_k + v_{cy}(x_k, y_k)\Big), \tag{13}$$

$$\psi_{k+1} = \psi_k + \Delta t\, r_k, \tag{14}$$

for $k = 0, \ldots, N-1$. The discretization process and the presence of control bounds can significantly affect the structure of the optimal solution. Moreover, the use of a soft terminal condition can produce multiple distinct trajectories with comparable travel time and path length, especially in regions where the current field exhibits symmetries, vortices, or singularities.

2.3.2. Discrete Optimization Problem

Based on previous content, the discrete optimization problem can be stated directly as

$$\begin{aligned} \min_{\{r_k\}_{k=0}^{N-1}, N} \quad & N\Delta t, \qquad N \in \mathbb{Z}^+ \\ \text{subject to} \quad & \mathbf{z}_{k+1} = f(\mathbf{z}_k, r_k, \Delta t), \quad k = 0, \ldots, N-1, \\ & \mathbf{z}_0 = \mathbf{z}_{\text{start}}, \\ & \left\| p_N - p_{\text{goal}} \right\|_2 \le \varepsilon, \\ & |r_k| \le r_{\max}, \quad k = 0, \ldots, N-1, \end{aligned} \tag{15}$$

where $f(\cdot)$ denotes the discrete Euler update in (12)–(14) and $\varepsilon > 0$ specifies the admissible final position tolerance.

The objective function in (15) minimizes the total travel time, which is proportional to the number of time steps N. The bound $|r_k| \le r_{\max}$ enforces the maximum admissible turning rate at each step, while the soft terminal constraint avoids requiring an exact match with the goal point. This formulation therefore preserves the physical feasibility of the heading-rate model, keeps the surge speed fixed as required by the Zermelo setting, and yields a discrete problem suitable for numerical optimization without introducing a separate exact-terminal formulation.

3. Hybrid Analytic/A*-Based Graph Search with Dynamic Target Generation

3.1. Overview of the Proposed Hybrid Search Strategy

The proposed method solves the Zermelo navigation problem through a hybrid framework that combines analytic local guidance with graph-based global exploration. Unlike conventional grid-based planners, the graph is constructed online in the continuous state space (x, y, ψ) by forward-simulating the vessel dynamics under the prescribed current field. Hence, every edge considered by the planner represents a dynamically feasible motion segment satisfying the constant surge-speed assumption and the imposed heading-rate bounds.

Local propagation is based on the analytical heading law of the uniform-current Zermelo problem in (7). In this work, however, this law is not used as a closed-form global solution. Instead, it acts as a low-level guidance rule that steers the vessel toward intermediate waypoint candidates generated during graph expansion. This is necessary

because the current field may be spatially varying and the implemented model uses the heading rate, rather than the heading angle, as the control input.

Global exploration is performed by a weighted A^* algorithm over adaptively quantized continuous states. Each node represents a vessel state, while each outgoing edge is generated by simulating the analytic guidance law toward a phase-dependent waypoint candidate. The edge cost is the navigation index of the simulated segment, so the search accounts jointly for travel time and path length. The heuristic is an optimistic lower bound based on the straight-line distance to the goal and the maximum possible ground speed.

To satisfy a prescribed execution-time constraint, the search is organized as a time-budgeted multi-phase procedure. Coarser phases use longer horizons, wider waypoint fans, and coarser quantization to promote exploration, whereas later phases refine the search around promising regions. An analytic baseline trajectory is computed before the graph search and used both as a reference candidate and as a fallback solution. The final trajectory is selected from the best complete candidates according to the navigation index, with best-effort and baseline recovery mechanisms used when necessary.

3.2. Local Node Expansion Through Analytic Propagation

The hybrid search procedure uses the analytical solution of the uniform-current Zermelo problem as a local propagation mechanism during graph expansion. Rather than defining successor nodes through geometric adjacency on a fixed lattice, the method generates them by simulating short feasible motion segments from the current search node toward a prescribed local target.

More specifically, let

$$\mathbf{z}_n = (x_n, y_n, \psi_n) \tag{16}$$

denote the state associated with the node currently being expanded, and let

$$z_t = (x_t, y_t) \tag{17}$$

denote a local target selected for propagation. Since the analytic expression in (7) is derived for uniform currents, the current field is evaluated locally at the current vessel position and treated as approximately constant over each short propagation step. The line-of-sight angle from the current position to the local target is then

$$\beta_t = \operatorname{atan2}(y_t - y_n,\ x_t - x_n) \tag{18}$$

At the same position, the local current magnitude and direction are given by

$$v_c(x_n, y_n) = \|\mathbf{v}_c(x_n, y_n)\|_2 \qquad \beta_c(x_n, y_n) = \operatorname{atan2}\big(v_{cy}(x_n, y_n),\ v_{cx}(x_n, y_n)\big) \tag{19}$$

Using the analytical law in (7) with the local target and the locally evaluated current, the reference heading for propagation is defined as

$$\psi_t^* = \beta_t - \arcsin\left(\frac{v_c(x_n, y_n)}{v_s}\sin(\beta_t - \beta_c(x_n, y_n))\right) \tag{20}$$

where v_s is the constant vessel speed. Because the dynamic model used in this work considers the heading rate as the control variable, the vessel heading is not set instantaneously equal to ψ_t^*. Instead, ψ_t^* is interpreted as a reference heading to be tracked through a bounded heading-rate command. The heading error is computed as

$$e_\psi = \operatorname{wrap}(\psi_t^* - \psi) \tag{21}$$

and the commanded heading rate is

$$r = \mathrm{clip}\left(k_p e_\psi,\, r_{\min},\, r_{\max}\right) \tag{22}$$

where $r_{\min}$ and $r_{\max}$ are the admissible turning-rate bounds. Thus, the analytical Zermelo expression provides the local desired direction, while the actual propagation remains consistent with the heading rate-constrained model.

In the implementation, the proportional gain is selected as

$$k_p = \frac{0.5}{\Delta t} \tag{23}$$

where Δt is the simulation discretization step. This choice is intended to exploit the maximum practical turning response allowed by the discrete propagation scheme while avoiding excessive oscillations or chattering in the control signal [34,35]. Therefore, the gain is not introduced as a free tuning parameter, but as a discretization-aware quantity that preserves a fast and numerically stable local steering behavior.

For each local target, the system is simulated over a short horizon determined by the active search phase and by the distance to the target. The terminal state of this rollout defines a candidate successor node, while the accumulated navigation index along the simulated segment defines the corresponding transition cost. If the propagated state does not yield effective progress with respect to the parent node, or if it remains equivalent to the parent under the active search resolution while the goal condition is not met, the transition is discarded.

Therefore, node expansion is performed through dynamically realizable motion segments guided by the same analytical heading relation used in the uniform-current Zermelo solution, but executed through the bounded heading-rate dynamics of the proposed model. This ensures that the local transitions used by the graph-search layer remain consistent with the vehicle dynamics, the constant-speed assumption, and the locally evaluated current field throughout the search process.

3.3. Dynamic Intermediate-Target Generation for Global Exploration

The local propagation mechanism described in the previous subsection requires a temporary target toward which the analytical guidance law is directed. If this target were restricted to the final destination, the method would behave as a purely local controller. To provide global exploration capability, the proposed solver generates a finite set of intermediate targets at each node expansion. For the node $\mathbf{z}_n = (x_n, y_n, \psi_n)$, with planar position $p_n = (x_n, y_n)$, the candidate set is written as

$$\mathcal{W}_n = \{p_{\mathrm{goal}}\} \cup \mathcal{W}_{\mathrm{ref},n} \cup \mathcal{W}_{\mathrm{fan},n} \tag{24}$$

where p_{goal} is always inserted first, $\mathcal{W}_{\mathrm{ref},n}$ contains optional targets extracted from an analytical reference trajectory, and $\mathcal{W}_{\mathrm{fan},n}$ is a phase-dependent polar fan. If p_n already satisfies the goal condition, only p_{goal} is retained.

The number of non-terminal target candidates retained at each expansion is adapted as a function of the current distance to the final target. More candidates are preserved near the goal, where finer directional resolution can improve the final approach, whereas fewer candidates are retained farther from the goal to reduce the branching factor and control the computational cost. After all candidates have been generated and filtered, the retained set includes only the selected non-terminal candidates and, when applicable, the terminal target.

The fan geometry is determined by the active search phase. Let T_h be the phase propagation horizon, R_g the goal radius, d_0 the initial start-to-goal distance, and $u_c^{\max}$ the maximum current magnitude. The characteristic spatial reach of the phase is defined as

$$L_{\text{reach}} = T_h \ \max(v_s + 0.35u_c^{\max},\ 0.75v_s) \tag{25}$$

and the waypoint scale is

$$s_w = \min\{d_0,\ \max(2R_g,\ L_{\text{reach}})\} \tag{26}$$

The normalized scale ratio is

$$\rho_s = \text{clip}\left(\frac{s_w}{d_0},\ 0.15,\ 1.0\right) \tag{27}$$

which defines the half aperture of the fan as

$$\phi = \text{clip}(20 + 40\rho_s,\ 18,\ 70)\ [\text{deg}] \tag{28}$$

The active radial fractions $\mathcal{R}$ and angular offsets $\mathcal{A}$ are selected as

ρ_s	$\mathcal{R}$	$\mathcal{A}$
$\rho_s \geq 0.70$	$\{0.35, 0.70, 1.00\}$	$\{0, -\phi/2, +\phi/2, -\phi, +\phi\}$
$0.35 \leq \rho_s < 0.70$	$\{0.45, 0.90\}$	$\{0, -\phi/2, +\phi/2, -\phi, +\phi\}$
$\rho_s < 0.35$	$\{0.55, 1.00\}$	$\{0, -\phi, +\phi\}$.

(29)

If lateral target generation is disabled, the angular set is reduced to $\mathcal{A} = \{0\}$.

Let

$$\gamma_n = \text{atan2}\left(y_{\text{goal}} - y_n,\ x_{\text{goal}} - x_n\right) \tag{30}$$

be the goal-oriented bearing from the current node. In standard phases, this bearing is used as the central fan direction. In current-guided adaptive phases, it is biased toward the local current direction as

$$\gamma_n^c = \gamma_n + \lambda_c \,\text{wrap}(\beta_c(x_n, y_n) - \gamma_n) \tag{31}$$

with $\lambda_c = 0.35$. Denoting by $\bar{\gamma}_n$ either γ_n or γ_n^c, each fan target is generated as

$$p_{\rho,\alpha} = p_n + r_{\rho,n}\begin{bmatrix}\cos(\bar{\gamma}_n + \alpha)\\ \sin(\bar{\gamma}_n + \alpha)\end{bmatrix} \quad \rho \in \mathcal{R},\ \alpha \in \mathcal{A} \tag{32}$$

where

$$r_{\rho,n} = \min\{d_n,\ \max(R_g,\ \rho s_w)\} \tag{33}$$

This construction distributes targets around the goal direction while adapting their range and angular spread to the current phase.

When analytical guidance is enabled and a valid baseline trajectory is available, additional reference targets are extracted from that trajectory. Let $\mathcal{B} = \{p_j^B\}_{j=0}^{N_B}$ denote the baseline positions. The closest baseline point to p_n is first identified, and M_B future samples are selected between that point and the end of the baseline, with $M_B = 4$ by default. Each selected point is inserted as a reference target. If lateral target generation is active, two additional targets are added along the normal direction to the local baseline tangent:

$$p_{j,\pm}^B = p_j^B \pm \ell_n \,\mathbf{n}_j^B \tag{34}$$

where $\mathbf{n}_j^B$ is a unit normal vector and

$$\ell_n = \min\{\lambda_B \max(0.75q_{xy},\ 0.60R_g),\ 0.15\max(d_n,\ R_g)\} \tag{35}$$

Here, q_{xy} is the active spatial quantization and λ_B is the analytical-guidance lateral scale, equal to 1.0 by default.

Candidate insertion includes a deduplication step. A new target w is merged with an existing candidate whenever their Euclidean distance is below

$$\varepsilon_{\text{dup}} = \max(0.25R_g,\ 0.50q_{xy}) \tag{36}$$

If the duplicated candidate corresponds to the true goal, the existing candidate is promoted to the first position and marked as terminal.

Before launching a rollout, each non-terminal candidate is filtered using a phase-dependent progress condition. With

$$\delta_{\text{phase}} = \max(1.25R_g,\ 0.25s_w) \tag{37}$$

a candidate w is retained only if

$$\left\|w - p_{\text{goal}}\right\|_2 \leq \left\|p_n - p_{\text{goal}}\right\|_2 + \delta_{\text{phase}} \tag{38}$$

This rule removes targets that move excessively away from the terminal region, while allowing limited detours needed to exploit favorable currents or avoid poor local flow structures. In specific adaptive phases, the prefilter can be disabled to recover wider exploration.

Finally, candidates that are too close to the parent node are not simulated. The rollout toward a target w is skipped if

$$\|w - p_n\|_2 \leq \max(0.25q_{xy},\ 0.10S_p) \tag{39}$$

where S_p is the estimated problem scale. If the remaining non-terminal candidate list exceeds $N_{\text{tar}}(d_n)$, it is linearly subsampled while preserving the terminal target. The retained targets are then passed to the analytic propagation step, and the resulting simulated terminal states define the successor nodes used by the global weighted-A^* search.

3.4. Weighted A Search over Dynamically Generated Motion Segments*

Once the local propagation stage has generated successor nodes toward dynamically selected targets, global exploration is organized through a weighted A^* procedure. The search graph is not available a priori. Instead, it is constructed online during the search, since the outgoing transitions of each expanded node are obtained only after generating candidate targets and simulating the corresponding motion segments.

Let n denote a search node associated with state $\mathbf{z}_n$, and let $\mathcal{N}(n)$ be the set of successor nodes produced by the local propagation mechanism. For each transition $n \to n'$, the edge cost is defined as the navigation index accumulated along the simulated rollout,

$$c(n, n') = NI(\tau_{n\to n'}), \tag{40}$$

where $\tau_{n\to n'}$ denotes the dynamically feasible segment connecting both nodes. Equivalently, using the navigation-index definition,

$$c(n, n') = T(\tau_{n\to n'}) + \frac{D(\tau_{n\to n'})}{v_s}. \tag{41}$$

The accumulated cost-to-come is updated whenever a lower-cost path to a successor is found,

$$g(n') \leftarrow g(n) + c(n, n'). \tag{42}$$

In the implementation, improved nodes may be reopened even if they have already been expanded, which allows the search to correct previous decisions when dynamically generated transitions provide a better accumulated cost.

Node prioritization is performed through the weighted A^* evaluation function

$$f(n) = g(n) + w_h h(n), \tag{43}$$

where $h(n)$ estimates the remaining cost-to-go and w_h is the heuristic weight. The default value is $w_h = 1$, which recovers the standard A^* evaluation rule, while larger values make the search more greedy with respect to the heuristic estimate.

The heuristic is expressed in navigation-index units. Let $p_n = (x_n, y_n)$ be the planar position of node n, and define

$$d_{\text{goal}}(n) = \left\| p_n - p_{\text{goal}} \right\|_2. \tag{44}$$

If $u_c^{\max}$ denotes the maximum current magnitude in the environment, the heuristic is given by

$$h(n) = \frac{d_{\text{goal}}(n)}{v_s + u_c^{\max}} + \frac{d_{\text{goal}}(n)}{v_s}. \tag{45}$$

The first term is an optimistic estimate of the remaining travel time, obtained by assuming the most favorable possible current contribution. The second term represents the corresponding distance contribution in the navigation-index metric. Thus, the heuristic is inexpensive to compute and is consistent with the cost used to evaluate simulated transitions.

The search is bounded by the maximum admissible navigation index associated with the prescribed simulation horizon. Let

$$T_{\max} = N_{\max} \Delta t \tag{46}$$

be the maximum simulated time. The corresponding cost budget is

$$J_{\max} = T_{\max} + T_{\max} \frac{v_s + u_c^{\max}}{v_s}. \tag{47}$$

This bound prevents the exploration from retaining paths whose accumulated cost is incompatible with the allowed time horizon.

A search phase terminates successfully when the node selected for expansion satisfies the goal condition. In the implemented procedure, a generated successor is not accepted immediately only because it lies in the goal region; the goal must be reached consistently through the search ordering. If the wall-clock budget of the current phase is exhausted before a complete path is found, the algorithm returns the best partial path available. Such partial solutions are ranked first by their remaining distance to the goal and, in case of comparable distances, by their accumulated navigation index.

Therefore, the weighted A^* layer determines how locally feasible motion segments are selected and concatenated. The propagation stage defines which transitions are physically executable, whereas the graph-search stage prioritizes those transitions according to accumulated navigation-index cost, optimistic remaining effort, admissible cost budget, and the available computation time.

3.5. Adaptive State Quantization and Duplicate Handling

The graph-search layer operates on states generated by continuous forward simulation. As a result, different rollouts may terminate at states that are numerically distinct while representing the same effective search situation. To avoid excessive growth of near-duplicate nodes, the proposed method identifies states through a phase-dependent quantized key. This key is used for node comparison, duplicate suppression, open- and closed-list bookkeeping, and transition validation, while the actual propagation remains continuous.

At the beginning of each search phase, the active spatial and angular resolutions are updated from the phase geometry. Let R_g be the goal radius, s_w the active waypoint scale, and S_p the estimated problem scale. The adaptive spatial resolution is computed as

$$q_{xy}^{\text{adapt}} = \max\left(0.10R_g,\, 0.08s_w,\, 0.02S_p\right) \tag{48}$$

The active spatial resolution is then obtained from the user parameter η_{xy} according to

$$q_{xy} = \begin{cases} q_{xy}^{\text{adapt}}, & \eta_{xy} \le 0 \\ \eta_{xy} q_{xy}^{\text{adapt}}, & 0 < \eta_{xy} \le 5 \\ \eta_{xy}, & \eta_{xy} > 5 \end{cases} \tag{49}$$

Thus, values of η_{xy} in $(0, 5]$ scale the adaptive resolution, whereas values larger than 5 are interpreted as absolute spatial resolutions.

The angular resolution is also phase dependent. Let ϕ denote the active fan half-aperture. The nominal angular resolution is computed as

$$q_{\psi}^{\text{adapt}} = \text{clip}\left(0.30\phi + \frac{180}{\pi}\arctan\left(\frac{q_{xy}}{s_w}\right),\, 2,\, 45\right) \tag{50}$$

The raw active value associated with the user parameter η_ψ is

$$\widetilde{q}_{\psi} = \begin{cases} q_{\psi}^{\text{adapt}}, & \eta_{\psi} \le 0 \\ \eta_{\psi} q_{\psi}^{\text{adapt}}, & 0 < \eta_{\psi} \le 5 \\ \eta_{\psi}, & \eta_{\psi} > 5 \end{cases} \tag{51}$$

and the final angular resolution is clipped as

$$q_{\psi} = \text{clip}\left(\widetilde{q}_{\psi},\, 2,\, 45\right) \tag{52}$$

This construction allows coarse phases to merge larger regions of the state space, while later phases can refine the angular and spatial resolution around promising regions.

For a continuous state $\mathbf{z} = (x, y, \psi)$, the quantized key is defined as

$$\kappa(\mathbf{z}) = \left(Q_x(x),\, Q_y(y),\, Q_\psi(\psi)\right) \tag{53}$$

where

$$Q_x(x) = \text{round}\left(\frac{x}{q_{xy}}\right) q_{xy} \tag{54}$$

$$Q_y(y) = \text{round}\left(\frac{y}{q_{xy}}\right) q_{xy} \tag{55}$$

and

$$Q_\psi(\psi) = \text{rad}\left(\text{round}\left(\frac{\deg(\psi) \bmod 360}{q_\psi}\right) q_\psi \bmod 360\right) \tag{56}$$

The heading is first mapped to $[0, 360)$ before rounding, so equivalent orientations separated by integer multiples of 2π are assigned to the same angular class.

Two propagated states that share the same key are treated as the same node for graph-search purposes. If a newly generated transition reaches an already represented key with a lower accumulated cost, the corresponding node can be updated and reopened by the search procedure. Otherwise, the duplicate transition is suppressed. This avoids redundant branching while preserving improved paths whenever they are discovered.

The same key is also used to validate simulated segments. Let $\mathbf{z}_n$ be the parent node and let $\mathbf{z}_{n'}$ be the terminal state of a rollout. If

$$\kappa(\mathbf{z}_{n'}) = \kappa(\mathbf{z}_n) \tag{57}$$

and the goal condition is not satisfied, the transition is discarded. This prevents the planner from spending search effort on rollouts that do not move the trajectory outside the active resolution cell.

Consequently, adaptive quantization provides a scale-aware mechanism for duplicate handling. It controls the effective size of the explored graph, stabilizes comparisons between continuously propagated states, and coordinates the graph-search resolution with the multi-phase exploration strategy.

3.6. Multi-Phase Search Under a Prescribed Execution-Time Budget

The proposed method is designed to operate under a prescribed maximum execution time. This is relevant because the difficulty of the navigation problem may vary significantly depending on the size of the domain, the current field, and the reachability of the goal.

To address this, the search is divided into several phases executed within the same global time budget. The analytic baseline is computed first, and the remaining time is used by the weighted A^* stages. Each phase uses its own exploration settings, such as the local propagation horizon, waypoint-generation geometry, filtering slack, and state quantization.

The first phases favor broader and coarser exploration, allowing the algorithm to cover large regions of the search space at moderate cost. Later phases use more selective settings to refine promising regions and improve the quality of the candidate solution.

If a complete solution is found, it is stored as the current best candidate. If no complete solution is available but the search continues to make progress and time remains, additional phases may be launched with modified parameters. In this way, the procedure remains adaptive while respecting the prescribed execution-time limit.

3.7. Final Solution Selection and Replay Validation

The final trajectory returned by the proposed method is not selected solely on the basis of graph-search completion. Instead, the algorithm compares the candidate solutions generated throughout the search process and retains the one that provides the most reliable outcome under the common navigation metric.

If one or more complete graph-search solutions are found, the best of them is identified according to its accumulated navigation-index cost. This candidate is then compared with the analytic baseline whenever the latter also reaches the goal successfully. If the baseline provides a lower navigation cost, it is retained as the final solution; otherwise, the best complete graph-search solution is selected. Therefore, the final output is not determined by the search architecture alone, but by the comparative quality of the feasible trajectories generated by the available solver components.

An additional validation stage is required because graph-search paths are represented internally as concatenations of local motion segments that must be replayed through the underlying dynamic model. For this reason, once a graph-search candidate has been

selected, its associated control history is reconstructed and re-simulated from the initial condition. The replayed trajectory is then checked against the terminal condition in order to verify that the selected graph-search solution remains dynamically consistent when executed as a complete control sequence. This prevents the algorithm from returning a graph-search path whose segment-wise construction appears feasible locally but does not reproduce a reliable goal-reaching trajectory when executed end to end. Consequently, the final selection stage combines cost comparison, replay verification, and fallback logic in order to improve robustness.

4. Benchmark Framework for the Zermelo Navigation Problem

To ensure consistent and reproducible evaluation of path-planning algorithms, a benchmark framework has been developed. This framework enables systematic specification of navigation scenarios through a set of key parameters defining the simulation environment, vessel dynamics, and termination conditions. Its modular design allows direct and fair comparison of different control strategies and search algorithms under identical conditions.

4.1. Scenario Parametrization

Each scenario is defined through a set of parameters that specify the simulation domain, initial and goal positions, heading-rate constraints, discretization step, and maximum time horizon. These parameters, summarized in Table 1, provide a unified representation of the navigation problem, scalable from small validation cases to large, ocean-like domains.

Table 1. Parameters defining the simulation environment and ship dynamics.

Parameter	Units	Description
`SIZE_ID`	integer [1,2]	1: (200×200) m^2 2: (2000×2000) m^2
`START`	m	Initial ship position (x, y)
`INITIAL_HEADING`	rad	Initial heading angle
`GOAL`	m	Target position (x, y)
`GOAL_RADIUS`	m	Goal radius (termination threshold)
`SHIP_VELOCITY`	m/s	Constant ship velocity
`CURRENT_VELOCITY`	m/s	Magnitude of current velocity
`R_MAX/R_MIN`	rad/s	Maximum/minimum heading rate
`DT`	s	Discretization time step
`MAX_TIME`	s	Maximum simulation time horizon
`MAX_EXECUTION_TIME`	s	Maximum solver runtime per scenario

To ensure reproducibility and fair comparison between algorithms, each parameter has been assigned either a randomization range or a computation rule, summarized in Table 2. This ensures that simulations remain statistically diverse while keeping solver results comparable across different scenario sizes.

A central element in this parametrization is the discretization step DT, which is scaled with the scenario size. This choice maintains a similar number of decision steps across scenarios, facilitating consistent evaluation of solution quality. However, increasing DT reduces temporal resolution and may overlook fine-grained dynamic effects, moving the problem away from exact real-world vessel behavior. For the basic Zermelo problem, this trade-off is considered acceptable, as the focus is on route planning performance rather than high-fidelity maneuvering.

The goal radius $GOAL_RADIUS$ is computed as

$$GOAL_RADIUS = 1.5 \times DT \times SHIP_VELOCITY,$$

ensuring that a solution can reach the goal within a single time step without overshooting. The heading-rate limit R_MAX directly constrains vessel maneuverability: smaller values may prevent sharp turns in strong currents, while larger values yield more agile trajectories.

Finally, $MAX_EXECUTION_TIME$ caps the runtime allowed per solver, ensuring that excessively slow solutions are discarded as impractical for real-time or large-scale use.

Table 2. Admissible parameter values and computation rules used in the benchmark scenarios.

Parameter	Units	Description
`INITIAL_HEADING`	rad	Random $\in (-\pi, \pi)$
`GOAL`	m	Random target position with distance $\in (0.75, 0.95) \times$ scenario size
`GOAL_RADIUS`	m	$1.5 \times$ DT $\times$ SHIP_VELOCITY
`SHIP_VELOCITY`	m/s	Random $\in (5.0, 15.0)$
`CURRENT_VELOCITY`	m/s	Random $\in (1.0, 4.0)$
`R_MAX`	rad/s	Random $\in (5.0, 15.0) \times \frac{\pi}{180}$
`R_MIN`	rad/s	– R_MAX
`DT`	s	Scenario 1: 0.5 s Scenario 2: 2.0 s
`MAX_TIME`	s	$5.0 \times \frac{\text{GOAL_DISTANCE}}{\text{SHIP_VELOCITY}}$
`MAX_EXECUTION_TIME`	s	Scenario 1: 3.0 s Scenario 2: 4.0 s

4.2. Performance Metrics

Solver evaluation is based on complementary performance indices that capture both trajectory quality and computational effort. The travel time measures the total duration to reach the goal, directly reflecting the degree of time-optimality. The travel distance quantifies the path length, indicating route efficiency and correlating with fuel consumption and operational cost. The execution time records the solver's computational effort, which is essential for real-time or large-scale applications.

To combine temporal and spatial efficiency of the trajectory solution a single scalar metric, a composite *Navigation Index* N_i is defined as

$$N_i = \texttt{total_time} + \frac{\texttt{total_distance}}{v_s} \tag{58}$$

where v_s is the nominal ship speed. Thus, the planner favors trajectories with low travel time, but penalizes solutions whose traveled distance grows excessively. This is useful because a purely time-optimal criterion may select routes that exploit favorable currents but introduce unnecessarily long detours. By adding the equivalent travel time associated with the path length, N_i provides a balanced measure that rewards fast trajectories while discouraging inefficient geometric behavior.

4.3. Current Fields

To evaluate the proposed methods under a broad range of flow conditions, the simulation framework incorporates several parameterizable current-field models representing qualitatively different navigation challenges. These fields range from simple baseline configurations, such as uniform or linearly sheared currents, to more structured and dynamically complex environments, including vortical flows, oscillatory tidal-like patterns, jet-like streams, wake structures, and pseudo-turbulent fields. The purpose of this diversity is to expose path-planning algorithms to different combinations of advection, local trapping, directional bias, and multi-scale flow variability, thereby enabling a more comprehensive assessment of their robustness, adaptability, and solution quality. Table 3 summarizes the current-field types currently implemented in the simulation environment.

Table 3. Available current field types in the simulation environment.

Current Field	Description
Uniform	Constant flow with fixed direction
Sinusoidal	Oscillatory flow with spatial variations in x and y
Logarithmic	Spiral flow with logarithmic radial decay
Gaussian swirl	Localized vortices with Gaussian decay
Vortex	Multiple vortices with random centers
Kármán vortex street	Alternating vortices forming a wake pattern
Coastal tidal	Oscillatory current with offshore decay
Linear shear	Velocity profile varying linearly across y
Double gyre	Two counter-rotating gyres in a bounded domain
Gaussian jet	Narrow, jet-like current with Gaussian profile
River outflow	Radial outflow with angular spreading and decay
Turbulence noise	Pseudo-turbulent field from random sine modes

5. Reference Solvers for Comparative Evaluation

To assess the quality of the trajectories produced by the proposed hybrid analytic/A^*-based planner, the benchmark framework incorporates several reference solvers. These methods are used not only to obtain alternative solutions, but also to evaluate trajectory quality, local improvability, and scalability across different scenario sizes. The reference solvers are grouped into two categories according to their role in the comparison: refinement-based solvers and independent reference solvers.

5.1. CasADi and GEKKO as Refinement-Based Solvers

The first group consists of CasADi and GEKKO, which are used as continuous optimization refiners initialized from the trajectory returned by the proposed planner. Their purpose is not to provide fully independent solutions, but to determine whether the A*-based trajectory can be locally improved. This makes them useful for assessing whether the proposed method already identifies high-quality solution regions.

These solvers are not forced to preserve the fixed benchmark time step Δt used in the simulation layer. Instead, they optimize a continuous-time trajectory representation with their own internal transcription or temporal parametrization. This allows them to explore smoother or temporally redistributed solutions and provides an indication of how close the proposed planner is to a strong local optimum.

Within this group, CasADi formulates the problem as a nonlinear program solved with IPOPT [36]. The state and control trajectories are decision variables, and the current field is incorporated through interpolated flow models evaluated during optimization. Candidate solutions can be initialized from the A*-derived trajectory and are validated through replay simulation before final selection.

GEKKO plays a similar role, but uses a simpler dynamic optimization formulation. It also optimizes final time and bounded heading-rate control, although the current field is represented through a profile built from the initial guess trajectory rather than through a fully state-dependent evaluation. Thus, GEKKO provides a lighter refinement mechanism, while CasADi offers a tighter continuous optimization model. Together, they provide complementary views of the local improvability of the proposed planner.

5.2. Pyomo and OMPL as Independent Reference Solvers

The second group consists of Pyomo [37,38] and OMPL. Unlike CasADi and GEKKO, these methods are not used merely to refine the proposed planner's output, but to search

for competitive trajectories through independent optimization or planning logic. They therefore act as stronger external references for evaluating the proposed method.

Pyomo provides an independent nonlinear optimal-control baseline. Its implementation uses a discrete-time model with midpoint integration of the current field, multi-seed initialization, adaptive control blocking, optional nonuniform time meshes, homotopy in current intensity, and replay-based validation. The pipeline combines feasibility-oriented stages with time-oriented refinement, making Pyomo a demanding benchmark capable of finding trajectories that may differ substantially from those produced by the hybrid A*-based planner.

OMPL, in contrast, provides a kinodynamic sampling-based baseline. It formulates the problem as a control-planning task over the Zermelo dynamics and uses planners such as KPIECE1, SST, RRT, and PDST to construct feasible trajectories. Since state propagation is performed under the local current field and the planner portfolio can adapt to the scenario scale, OMPL is especially relevant in large or geometrically complex domains.

5.3. Comparative Protocol and Role Within the Benchmark

The comparative protocol is deliberately favorable to the reference solvers. The proposed planner starts from scratch, uses the fixed simulation step Δt, and operates under a more restrictive execution-time budget. In contrast, CasADi and GEKKO are initialized from the proposed trajectory, are not constrained by the same fixed discretization, and are allowed additional computation time. Similarly, Pyomo and OMPL use independent formulations but are also executed with a larger time budget.

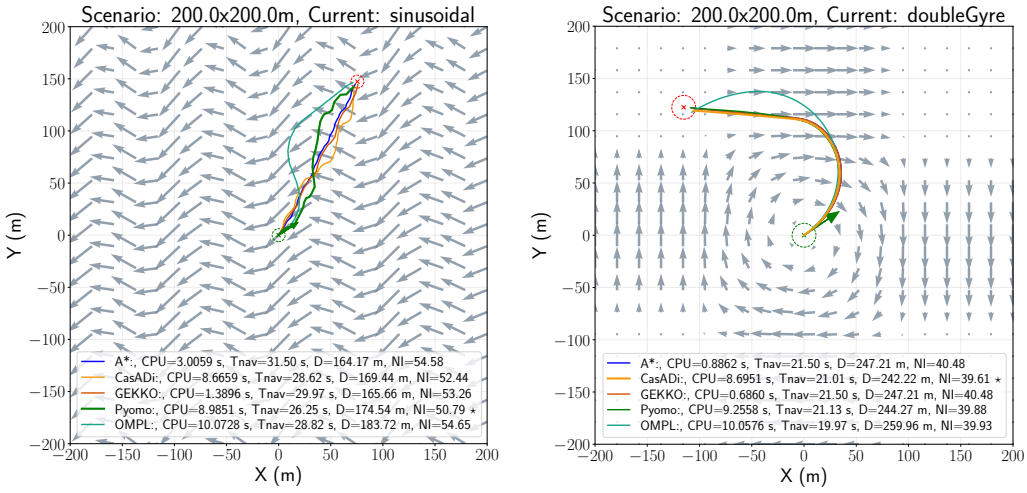
This asymmetric setup is intentional: it evaluates whether the proposed planner can produce solutions that remain close to those obtained under more favorable optimization conditions. If CasADi or GEKKO significantly improve the proposed trajectory, then the planner may be identifying a promising region without reaching a strong local optimum. If Pyomo or OMPL consistently find better independent solutions, then relevant trajectory families may exist outside the regions explored by the proposed search strategy.

Conversely, if the proposed hybrid planner remains competitive against both refinement-based and independent solvers, this supports its ability to combine global exploration with strong local trajectory quality. The comparison therefore evaluates not only raw performance, but also robustness, adaptability, and scalability under varying scenario size, current-field structure, and execution-time constraints.

6. Structural Challenges of the Zermelo Navigation Problem

The Zermelo navigation problem becomes substantially more difficult when realistic flow fields, bounded turning dynamics, and large spatial domains are considered. Its complexity arises from the interaction of several structural features that reshape the feasible trajectory set and make global solution quality difficult to guarantee. While uniform-current cases often lead different solvers to nearly identical trajectories, non-uniform and nonlinear currents produce richer, multimodal solution spaces where local guidance alone may be insufficient.

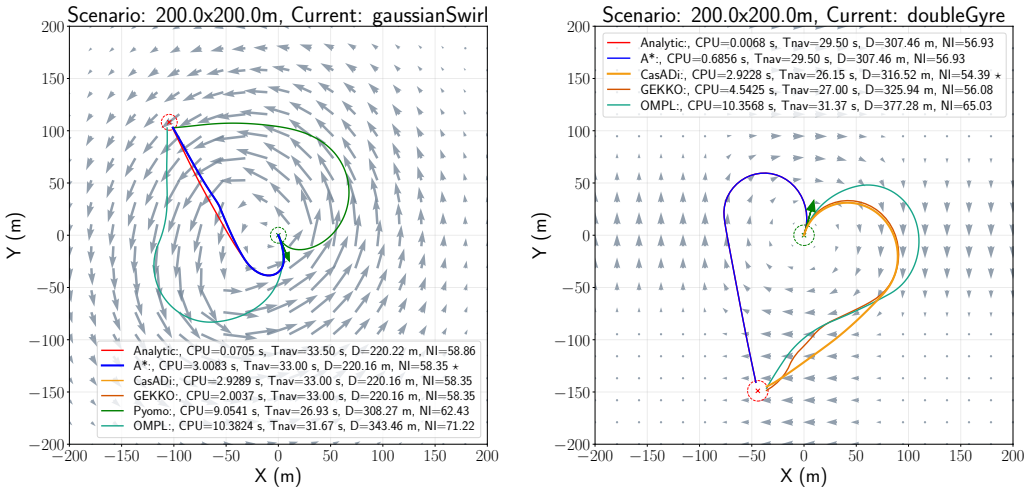
1. **Multiplicity of locally competitive solutions.** The problem may admit several qualitatively different trajectories with similar navigation index values, as shown in Figure 1. In the sinusoidal current field, spatially distinct paths lead to nearly equivalent performance. In the double gyre case, the OMPL trajectory differs substantially from the others: it travels a longer distance but achieves a shorter travel time, resulting in a comparable navigation index. These cases show that different time–distance trade-offs may be locally competitive, making the optimization landscape intrinsically multimodal.



(**a**) Sinusoidal current field. (**b**) Double gyre current field.

Figure 1. Examples of multiple locally competitive solutions. Spatially distinct trajectories may yield similar navigation index values because longer paths can be compensated by shorter travel times.

2. **Effect of symmetries and quasi-symmetries.** Symmetries or near-symmetries in the flow field may generate mirrored trajectories with comparable objective values, as illustrated in Figure 2. Although these paths differ geometrically, their navigation performance can remain nearly equivalent. As a result, several branches of the solution space may appear similarly promising, making the final trajectory more sensitive to the exploration strategy, initialization, and computational budget.



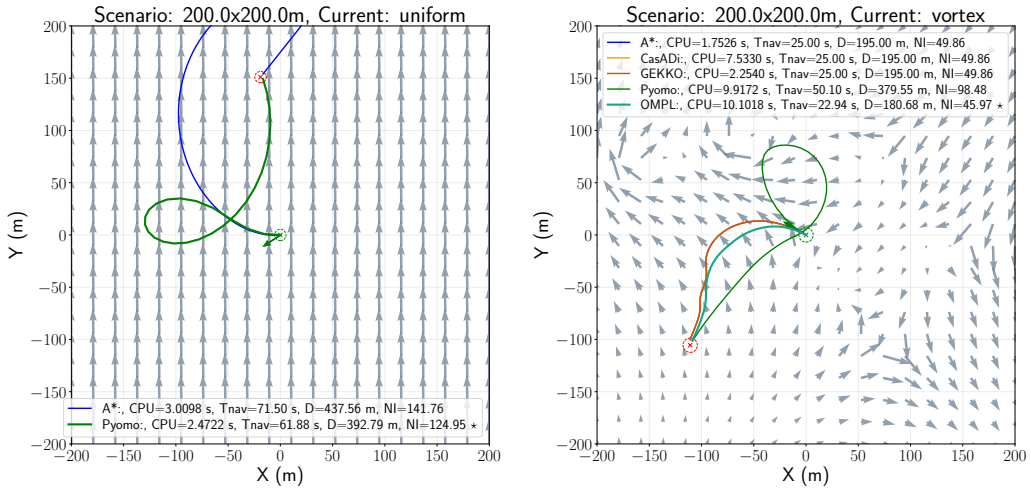
(**a**) Gaussian swirl current field. (**b**) Double gyre current field.

Figure 2. Symmetric or near-symmetric solutions in different flow fields. Several solvers produce mirrored trajectories with comparable navigation performance.

3. **Conflict between local optimality and turning-rate constraints.** When the heading-rate constraint $|r| \leq r_{\max}$ is enforced, the classical analytical solution based on direct heading control may no longer be feasible or globally competitive. Even in uniform currents, the vessel may be unable to reorient fast enough toward the locally optimal direction, producing wide turns or sliding arcs. Thus, locally optimal headings do not necessarily generate globally efficient feasible trajectories.
4. **Presence of singularities and sharp flow transitions.** Nonlinear current fields may contain singular points, sharp gradients, vortices, or regions where the flow structure

changes abruptly. These features distort the reachable set and may force trajectories to avoid adverse-flow regions or exploit narrow favorable corridors. As shown in Figure 3, such structures can also induce inefficient maneuvers, such as unnecessary loops, when the solver interacts poorly with the local flow geometry.

5. **Growth of the search space with domain scale.** As the spatial domain increases, the number of relevant trajectory families grows rapidly. Long routes may exploit distant favorable currents, avoid adverse regions through large detours, or reveal symmetries that are not visible at smaller scales. Therefore, large-domain instances are not only quantitatively larger, but also qualitatively harder, requiring both global exploration and feasible computation time.
6. **Emergence of loops and partially redundant trajectories.** Feasible trajectories may exhibit loops or partially redundant segments due to bounded turning-rate dynamics. Figure 3 shows two representative cases. In the left panel, a loop-like maneuver may be beneficial because it allows the vehicle to approach the target with a more favorable heading. In the right panel, the loop generated by Pyomo increases path length and travel time, degrading the solution. Hence, loop emergence is not intrinsically good or bad; its effect depends on its interaction with maneuverability constraints, current structure, and route geometry.



(**a**) Uniform current field. (**b**) Vortex current field.

Figure 3. Loop-like trajectories under bounded turning-rate dynamics. A loop may improve the final approach direction, but it may also introduce unnecessary distance and travel time depending on the flow and route geometry.

7. **Sensitivity to discretization and termination criteria.** The computed solution may depend strongly on the time step, stopping tolerance, and solver-specific convergence criteria. Fine discretizations improve fidelity but increase computational cost, whereas coarse discretizations may miss important dynamic effects. This sensitivity is especially relevant when comparing fixed-step simulation methods with solvers using nonuniform or variable temporal parametrizations.

Taken together, these features explain why the Zermelo navigation problem remains difficult even with relatively simple vessel dynamics. The main challenge is not only to find a feasible path, but to identify a trajectory that remains competitive in both travel time and path efficiency while respecting turning-rate constraints. This motivates hybrid methods that combine global exploration with dynamically meaningful local propagation.

7. Numerical Results and Comparative Evaluation

This section evaluates the proposed hybrid *A**-based solver on 2400 Zermelo navigation scenarios, corresponding to 12,000 solver runs across two scenario sizes and twelve current-field families. The comparison includes *A**, CasADi, GEKKO, Pyomo, and OMPL, using success rate, best-solver frequency, and Navigation Index (NI) as the main performance metrics.

As discussed previously, CasADi and GEKKO are used as A*-initialized refinement methods, whereas Pyomo and OMPL act as independent reference solvers. The protocol is deliberately favorable to the reference methods: they are given larger computational budgets, and the refinement solvers start from an already feasible A* trajectory. Therefore, competitive A* performance under these conditions indicates bounded-time robustness and practical trajectory quality, rather than the outcome of a symmetric comparison.

7.1. Global Benchmark Performance

Table 4 summarizes the aggregate results over the full benchmark. All solvers are evaluated on 2400 scenarios. The proposed *A** solver reaches the goal in 2335 cases, corresponding to a success rate of 97.292%. This places it in the same high-robustness range as CasADi, GEKKO, and OMPL, whose success rates are all above 96%. In contrast, Pyomo succeeds in 1703 cases, with a substantially lower success rate of 70.958%.

The last two columns report how often each solver attains the lowest Navigation Index (NI) among the successful solutions for a scenario. For the refinement-based solvers, ties are assigned to *A**: if CasADi or GEKKO only reproduces the A*-initialized trajectory, it is not counted as an improvement. Under this convention, *A** obtains the best NI in 445 scenarios, while Pyomo and OMPL obtain larger best-count values, with 788 and 525 cases, respectively.

These results separate two aspects of performance. The proposed *A** method is highly reliable under a bounded-time setting, whereas some reference solvers can obtain lower NI values in subsets of scenarios when they converge or when they are run under more favorable computational conditions.

Table 4. Global solver performance over the full benchmark.

Solver	Runs	Successes	Success (%)	Best Count	Best Share (%)
A*	2400	2335	97.292	445	18.542
CasADi	2400	2349	97.875	361	15.042
GEKKO	2400	2360	98.333	281	11.708
Pyomo	2400	1703	70.958	788	32.833
OMPL	2400	2326	96.917	525	21.875

7.2. Robustness Across Scenario Size and Current Regime

Figure 4 reports success rates by solver, scenario size, and current type. The proposed A* solver remains consistently robust across the benchmark, with a best cell of 100% success for size-2 `coastalTidal` scenarios and a worst cell of 89% for size-1 `gaussianSwirl` scenarios. CasADi and GEKKO show similar robustness patterns, with worst-cell success rates of 92% and 93%, respectively. OMPL also remains globally robust, with a worst-cell success rate of 90%. Pyomo displays a much larger variability: its best cell reaches 87% success for size-2 `coastalTidal` scenarios, whereas its worst cell drops to 4% for size-2 `sinusoidal` scenarios.

This robustness map is important because it separates feasibility from conditional trajectory quality. A solver that obtains low NI values only on a reduced subset of successful

runs may still be less useful as a general-purpose planner than a solver with slightly higher NI but much broader coverage. Under this view, A* is not only strong in aggregate, but also stable across heterogeneous current families and problem scales. This behavior is consistent with the intended design of the proposed method: analytic local propagation, dynamic target generation, time-budgeted multi-phase search, and robust final selection produce a planner that is not tied to a single operating regime.



Figure 4. Success-rate tables by solver, scenario size, and current-field family. Each cell reports the percentage of scenarios in which the corresponding solver reaches the goal for a given size–current pair. The figure should be read as a robustness map rather than as a trajectory-quality comparison.

7.3. A as a Seed Generator for CasADi and GEKKO Refinement*

The comparisons with CasADi and GEKKO must be read as refinement analyses. Since both solvers are initialized from the A* trajectory, their role is to determine how much additional improvement can be extracted from an A*-generated solution when extra continuous-optimization effort is applied after the bounded-time search.

Against CasADi, there are 2400 comparable scenarios and a beat count of 816 (34.00%), composed of 802 jointly successful cases with better NI and 14 CasADi-only successes. Against GEKKO, the beat count is 287 (11.96%), composed of 262 jointly successful NI improvements and 25 GEKKO-only successes. In both comparisons, the number of jointly successful cases in which the refinement solver returns a worse NI than A* is zero.

However, the improvement counts must be interpreted together with their magnitude. On jointly successful scenarios, the median NI improvement over A* is 0.000% for both CasADi and GEKKO. The mean NI improvement is 0.591% for CasADi and 0.153% for GEKKO. Thus, the main conclusion is not that refinement produces large systematic gains, but rather that the A* trajectory is already very competitive in most cases, with downstream optimizers extracting only modest additional benefit on average. This interpretation is supported by Figure 5, whose left panel shows that most refinement gains are concentrated near zero, with only a limited tail of larger improvements.

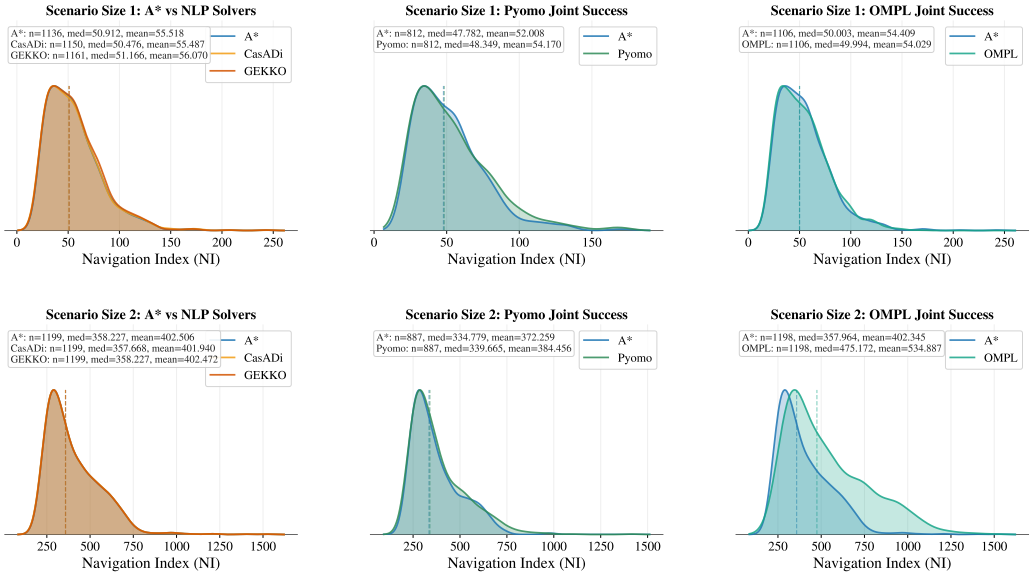


Figure 5. Quantitative comparison of A* with refinement-based solvers and independent reference solvers. The left panel shows that CasADi and GEKKO extract mostly small NI improvements from the A* seed, while the right panel shows that Pyomo can improve NI in some converged cases and OMPL generally yields lower trajectory quality.

The size-wise breakdown further clarifies the role of A* as a seed generator. The CasADi beat share is 46.00% for size 1 and 22.00% for size 2. For GEKKO, the corresponding values are 22.42% and 1.50%. Therefore, A* frequently provides a trajectory that can be locally polished, especially in smaller scenarios, but the central tendency of the improvement remains close to zero. In larger scenarios, the lower beat share of the refiners indicates that the A* seed is often already close to the local solution recovered by these downstream formulations, although isolated improvements remain possible.

7.4. Independent Reference Solvers: Pyomo and OMPL

The comparisons with Pyomo and OMPL address a different question. Unlike CasADi and GEKKO, these solvers do not refine the A* solution. They search independently

for feasible trajectories using alternative formulations and larger computational budgets. Their role is therefore to test whether solution families exist beyond those found by the bounded-time A*-based planner.

Pyomo reveals a clear trade-off between robustness and conditional quality. Globally, Pyomo succeeds in 1703 out of 2400 scenarios, far below the 2335 successes obtained by A*. This gap is also visible in the pairwise outcome table: A* has 636 A*-only successes against Pyomo, whereas Pyomo has only 4 solver-only successes. Nevertheless, on jointly successful scenarios, Pyomo often finds lower-NI trajectories. It improves on A* in 1110 jointly successful cases, corresponding to 46.250% of all comparable scenarios, while it is worse than A* in 589 cases (24.542%). The relative NI statistics are mixed: the median improvement is $+0.666\%$, whereas the mean improvement is -3.561%, indicating unfavorable tails. Pyomo can therefore discover trajectories that improve on A* when it converges, but this advantage is offset by much weaker reliability.

OMPL presents a different pattern. Its global success rate, 96.917%, is close to that of A*, but its conditional NI quality is generally worse in this benchmark. On jointly successful scenarios, OMPL improves on A* in 583 cases (24.292% of all comparable scenarios), whereas it is worse than A* in 1721 cases (71.708%). The median relative NI improvement is -10.116% and the mean is -16.294%, both under the convention that positive values indicate lower NI than A*. The degradation is especially pronounced for size-2 scenarios, where the median relative NI improvement is -29.20%. Hence, although OMPL is a meaningful independent kinodynamic-planning baseline and remains robust in terms of success rate, it does not generally outperform A* in trajectory quality on this benchmark.

The right panel of Figure 5 visualizes these independent comparisons. For Pyomo, a substantial fraction of jointly successful cases lies below the parity diagonal, reflecting its ability to find lower-NI trajectories in many scenarios. For OMPL, most points lie above the diagonal, consistent with its negative median and mean relative improvement values. These results show that independent search can reveal alternative trajectories, but they also confirm that the proposed A*-based planner remains strongly competitive under a more restrictive computational budget.

7.5. Distributional View of Navigation Index

Figures 6 and 7 provide a distributional view of NI by current type and scenario size. These figures include only successful runs, and therefore must be interpreted together with the success-rate information in Figure 4 and Table 4.

For size 1 (200×200 m), the median NI values of A*, CasADi, GEKKO, and OMPL are tightly clustered: A* obtains 50.912, CasADi 50.476, GEKKO 51.166, and OMPL 50.359. Pyomo attains the lowest median NI, 48.220, but only over 816 successful samples, compared with 1136 for A*, 1150 for CasADi, 1161 for GEKKO, and 1127 for OMPL. Thus, Pyomo's lower conditional median should not be interpreted independently of its lower success coverage.

For size 2 (2000×2000 m), A*, CasADi, and GEKKO remain nearly indistinguishable in central tendency, with median NI values of 358.227, 357.668, and 358.227, respectively. Pyomo again shows a lower median NI, 339.665, but over only 887 successful samples. OMPL exhibits a substantially larger median NI of 475.579, despite achieving 1199 successful samples. This confirms the pattern observed in Figure 5: OMPL remains robust in feasibility terms, but its trajectories are typically less competitive in NI for the larger scenarios.

Taken together, the distributional figures support two conclusions. First, the proposed A* solver produces trajectories whose central NI statistics are very close to those obtained by the A*-initialized refinement solvers. This is consistent with the interpretation that A* already provides strong seed trajectories before downstream optimization. Second,

conditional trajectory quality and solver coverage must be evaluated jointly. A lower NI distribution over successful cases does not necessarily imply better global benchmark behavior when the same solver has a much lower success rate.

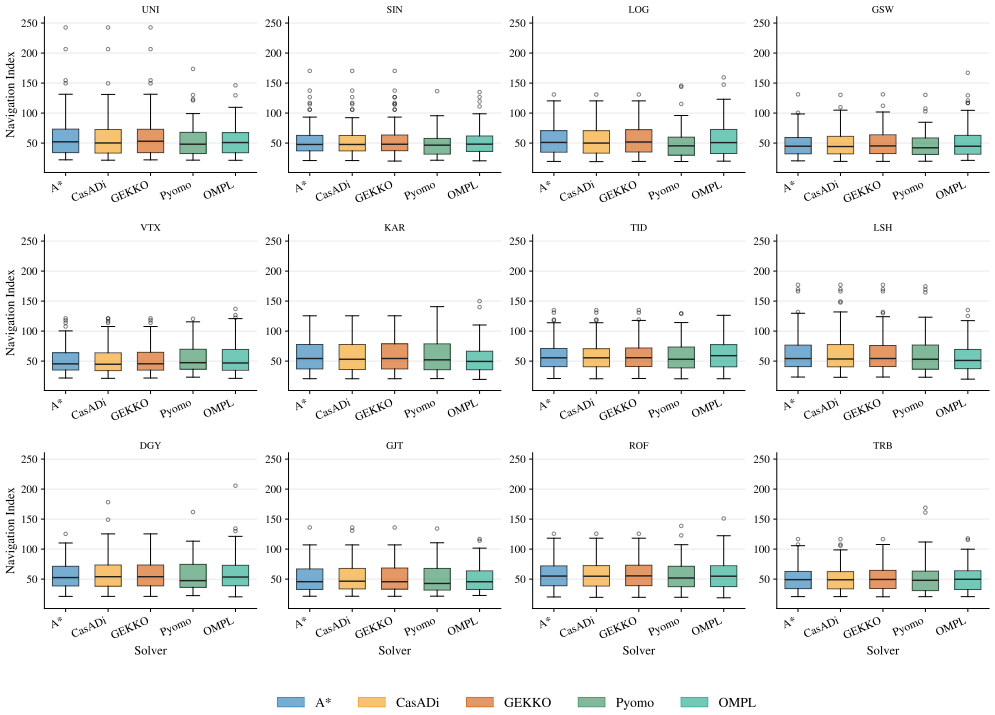


Figure 6. Distribution of Navigation Index values for successful runs in size-1 scenarios, grouped by current-field family and solver. Since failed runs are excluded from the boxplots, the figure represents conditional trajectory quality rather than overall benchmark reliability. For size 1, A*, CasADi, GEKKO, and OMPL have closely clustered median NI values, while Pyomo attains a lower median NI on a smaller successful subset.

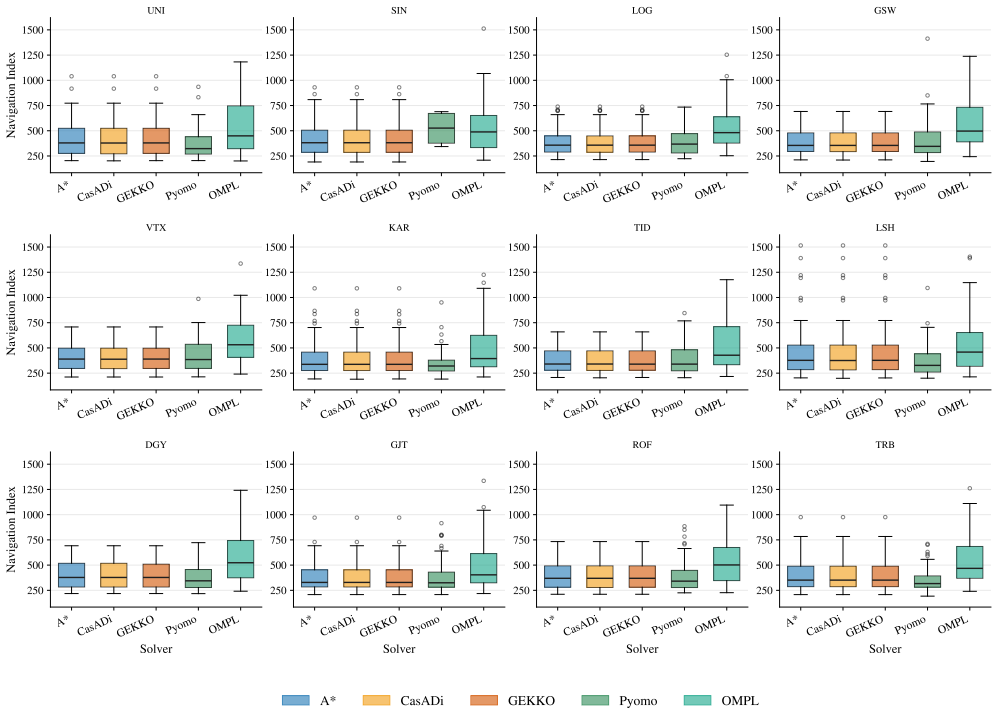


Figure 7. Distribution of Navigation Index values for successful runs in size-2 scenarios, grouped by current-field family and solver. The figure highlights the near-identical central NI behavior of A*,

CasADi, and GEKKO after successful completion, the lower conditional median achieved by Pyomo on fewer successful runs, and the larger NI values obtained by OMPL in the larger scenarios.

7.6. Overall Assessment

This work presented a hybrid analytic/A^*-based planner for the Zermelo navigation problem in non-uniform current fields with bounded heading-rate dynamics. The method combines analytical local propagation with online graph construction, where each edge is generated by simulating a dynamically feasible motion segment toward an intermediate target. Thus, the planner avoids a fixed spatial lattice and searches directly over motion segments consistent with the vehicle dynamics and the locally evaluated current field.

The algorithm was designed to balance trajectory quality, robustness, and bounded computation time. Dynamic target generation expands the search beyond purely goal-directed guidance, while weighted A^* organizes exploration according to the Navigation Index. The analytic baseline, multi-phase time-budgeted search, adaptive state quantization, robust final selection, and replay validation all contribute to reliable behavior across spatial scales and flow structures.

The benchmark results over 2400 scenarios and 12,000 solver runs show that the proposed method reaches a high success rate of 97.292%. This places it close to the strongest reference methods in feasibility, namely CasADi, GEKKO, and OMPL, while using a more restrictive computational budget. Its success-rate profile remains stable across scenario sizes and current-field families, indicating robust bounded-time behavior rather than specialization to a narrow regime. Although Pyomo obtains lower Navigation Index values in many jointly successful cases, its lower success rate limits its reliability. OMPL is also robust in feasibility, but generally produces higher Navigation Index trajectories, especially in size-2 scenarios.

The refinement experiments with CasADi and GEKKO show that the trajectories returned by the proposed planner are often already close to a local optimum. Both solvers improve some A^*-initialized trajectories, but the median Navigation Index improvement is zero and the mean improvements remain small. This indicates that substantial local improvements are not systematically available after refinement, and that the planner often provides high-quality solutions before any downstream continuous optimization.

Overall, the results support the main hypothesis of this work: combining analytical local guidance with graph-based global exploration yields a practical compromise between bounded-time feasibility, dynamic consistency, broad applicability, and competitive trajectory quality. The proposed framework can therefore be interpreted as a robust core planner, usable directly under strict computation-time constraints or as a provider of high-quality initial trajectories for continuous-optimization methods when more computation time is available.

Several limitations remain. The heuristic relies on a global bound on current magnitude and does not exploit detailed local flow structure. Performance also depends on waypoint generation, prefiltering, and phase-dependent quantization parameters. Moreover, simulating local neighbors can become expensive in large domains. Future work should investigate current-aware heuristics, adaptive waypoint policies, and more efficient strategies for pruning or reusing local rollouts.

8. Discussion

The numerical results indicate that current-aware marine route planning cannot be reduced to a purely geometric shortest-path problem. In non-uniform flow fields, trajectory quality depends on the interaction between the vehicle motion constraints and the spatial structure of the current. This is particularly relevant for autonomous marine vehicles,

where the environment may either assist or oppose the commanded motion depending on the selected route.

The proposed planner addresses this issue by combining local dynamically feasible propagation with graph-based exploration. This design provides a compromise between purely continuous optimal-control solvers and conventional graph-search methods. Unlike fixed-lattice approaches, the search is built from motion segments that are generated online and evaluated using the local current field. At the same time, the use of a weighted A^* structure keeps the exploration organized and computationally bounded.

The benchmark study suggests that this hybrid structure is especially useful when robustness and execution time are important. The method achieves a high feasibility rate across different scenario sizes and current-field families, while maintaining competitive Navigation Index values. This behaviour supports its use as a practical core planner in settings where repeated planning calls are required, such as simulation campaigns, digital-twin environments, or onboard decision-support systems.

The refinement experiments also provide useful insight into the role of the proposed planner within a broader optimization pipeline. Since continuous solvers such as CasADi and GEKKO produce only limited average improvements when initialized with the proposed trajectories, the planner appears to return solutions that are already close to locally optimized routes in many cases. Therefore, the method can be used either as a standalone bounded-time planner or as an initializer for more expensive continuous-optimization procedures.

Several limitations remain. The heuristic currently relies on a global bound on the current magnitude and does not fully exploit local flow structures. The quality and computational cost of the search also depend on the waypoint-generation strategy, the prefiltering rule, and the state-quantization parameters. In larger domains, the repeated simulation of local motion segments may become expensive. Future work should therefore investigate more informative current-aware heuristics, adaptive waypoint policies, rollout reuse, and extensions involving forecast uncertainty, energy-aware costs, dynamic obstacles, and COLREG-compliant navigation.

9. Conclusions

This work introduced a hybrid analytic/A^*-based planner for the Zermelo navigation problem in non-uniform current fields with bounded heading-rate dynamics. The method provides dynamically feasible, current-aware trajectories without relying on a fixed spatial lattice.

The results show that the proposed approach offers a robust balance between feasibility, computational efficiency, and trajectory quality over a broad benchmark set. Its performance, together with the limited gains obtained from subsequent continuous refinement, indicates that the planner is suitable both for direct bounded-time use and for initializing downstream optimal-control solvers.

Overall, the proposed framework provides a practical basis for autonomous marine route planning in flow-driven environments. Future developments will focus on improving flow-aware guidance, reducing rollout costs, and incorporating more realistic operational constraints.

10. Methods

During the preparation of this manuscript, ChatGPT (GPT-5.5 Instant) was used as an AI-assisted copy-editing tool to improve the English language of human-written text. Its use was limited to spelling and grammar correction, improvement of readability and style, and semantic clarification of the wording in English.

All AI-assisted edits were reviewed, revised, and validated by the author. The author confirms that the final version of the manuscript reflects its original work. To the best of the author's knowledge, and based on the literature reviewed, no previous work has been found that develops the same methodological approach for the problem considered in this study. The present work therefore proposes an original solution, grounded in a critical analysis of related research.

Funding: J. A. González-Prieto gratefully acknowledges funding from the own research funding programme of the Centro Universitario de la Defensa en la Escuela Naval Militar, through the research project "Plataforma de ensayo y validación de un gemelo digital oceanográfico (POSEIDON)" (PICUD-2023-01).

Data Availability Statement: Data sharing is not applicable to this article as no external datasets were generated or analyzed during the current study. The software utilized in this work is currently not publicly available due to institutional restrictions.

Conflicts of Interest: The author declares that there is no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

A*	A-star
AUV	Autonomous Underwater Vehicle
CPU	Central Processing Unit
DWA	Dynamic Window Approach
GEKKO	Optimization suite for dynamic systems and nonlinear programming
IPOPT	Interior Point OPTimizer
NI	Navigation Index
NLP	Nonlinear Programming
OMPL	Open Motion Planning Library
PRM	Probabilistic Roadmap
RRT	Rapidly-exploring Random Tree
SST	Stable Sparse RRT
USV	Unmanned Surface Vehicle

References

1. Zermelo, E. Über das Navigationsproblem bei ruhender oder veränderlicher Windverteilung. *ZAMM-J. Appl. Math. Mech./Z. Angew. Math. Mech.* **1931**, *11*, 114–124. [CrossRef]
2. Bryson, E.A. *Applied Optimal Control: Optimization, Estimation and Control*; Routledge: Abingdon, UK, 2018.
3. Techy, L. Optimal navigation in planar time-varying flow: Zermelo's problem revisited. *Intell. Serv. Robot.* **2011**, *4*, 271–283. [CrossRef]
4. Li, B.; Xu, C.; Teo, K.L.; Chu, J. Time optimal Zermelo's navigation problem with moving and fixed obstacles. *Appl. Math. Comput.* **2013**, *224*, 866–875. [CrossRef]
5. Biferale, L.; Bonaccorso, F.; Buzzicotti, M.; Clark Di Leoni, P.; Gustavsson, K. Zermelo's problem: Optimal point-to-point navigation in 2D turbulent flows using reinforcement learning. *Chaos Interdiscip. J. Nonlinear Sci.* **2019**, *29*, 103138. [CrossRef] [PubMed]
6. Jeong, S.; Kim, T.-W. Generating a path-search graph based on ship-trajectory data: Route search via dynamic programming for autonomous ships. *Ocean Eng.* **2023**, *283*, 114503. [CrossRef]
7. Mannarini, G.; Subramani, D.N.; Lermusiaux, P.F.; Pinardi, N. Graph-search and differential equations for time-optimal vessel route planning in dynamic ocean waves. *IEEE Trans. Intell. Transp. Syst.* **2019**, *21*, 3581–3593. [CrossRef]
8. Singh, Y.; Sharma, S.; Sutton, R.; Hatton, D.; Khan, A. A constrained A* approach towards optimal path planning for an unmanned surface vehicle in a maritime environment containing dynamic obstacles and ocean currents. *Ocean Eng.* **2018**, *169*, 187–201. [CrossRef]
9. Yang, C.; Pan, J.; Wei, K.; Lu, M.; Jia, S. A novel unmanned surface vehicle path-planning algorithm based on A* and artificial potential field in ocean currents. *J. Mar. Sci. Eng.* **2024**, *12*, 285. [CrossRef]

10. Mu, D.; Li, T.; Han, X.; Fan, Y.; Wang, F. Global path planning for unmanned surface vehicles in complex maritime environments considering environmental interference. *Ocean Eng.* **2024**, *310*, 118580. [CrossRef]
11. Liu, H.; Shan, Q.; Cao, Y.; Xu, Q. Global path planning of unmanned surface vehicle in complex sea areas based on improved streamline method. *J. Mar. Sci. Eng.* **2024**, *12*, 1324. [CrossRef]
12. Zhu, T.; Xiao, Y.; Zhang, H. Path planning of USV based on improved PRM under the influence of ocean current. *Proc. Inst. Mech. Eng. Part M J. Eng. Marit. Environ.* **2024**, *238*, 954–967. [CrossRef]
13. Zaccone, R.; Martelli, M. A random sampling based algorithm for ship path planning with obstacles. In Proceedings of the ISCSS, Glasgow, UK, 2–4 October 2018; Volume 2018.
14. Zaccone, R.; Martelli, M.; Figari, M. A colreg-compliant ship collision avoidance algorithm. In Proceedings of the 2019 18th European Control Conference (ECC), Naples, Italy, 25–28 June 2019; pp. 2530–2535.
15. Zaccone, R.; Martelli, M. A collision avoidance algorithm for ship guidance applications. *J. Mar. Eng. Technol.* **2020**, *19*, 62–75. [CrossRef]
16. Krell, E.; King, S.A.; Carrillo, G.L.R. Autonomous Surface Vehicle energy-efficient and reward-based path planning using Particle Swarm Optimization and Visibility Graphs. *Appl. Ocean Res.* **2022**, *122*, 103125. [CrossRef]
17. Han, Q.; Yang, X.; Song, H.; Sui, S.; Zhang, H.; Yang, Z. Whale optimization algorithm for ship path optimization in large-scale complex marine environment. *IEEE Access* **2020**, *8*, 57168–57179. [CrossRef]
18. Akbari, B.; Pan, Y.-J.; Liu, S.; Wang, T. Optimized Trajectory Planning for USVs Under Ocean Currents. *IEEE J. Ocean. Eng.* **2025**, *50*, 2661–2671. [CrossRef]
19. 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*, 61–72. [CrossRef]
20. Song, Z.; Zhang, J.; Wu, D.; Tian, W. A novel path planning algorithm for ships in dynamic current environments. *Ocean Eng.* **2023**, *288*, 116091. [CrossRef]
21. Panigrahi, J.K.; Padhy, C.P.; Sen, D.; Swain, J.; Larsen, O. Optimal ship tracking on a navigation route between two ports: A hydrodynamics approach. *J. Mar. Sci. Technol.* **2012**, *17*, 59–67. [CrossRef]
22. Li, G.; Hildre, H.P.; Zhang, H. Toward time-optimal trajectory planning for autonomous ship maneuvering in close-range encounters. *IEEE J. Ocean. Eng.* **2019**, *45*, 1219–1234. [CrossRef]
23. Wen, J.; Dai, H.; He, J.; Sun, L.; Gao, L. Intelligent decision-making method for AUV path planning against ocean current disturbance via reinforcement learning. *IEEE Internet Things J.* **2024**, *11*, 38965–38975. [CrossRef]
24. Yoo, B.; Kim, J. Path optimization for marine vehicles in ocean currents using reinforcement learning. *J. Mar. Sci. Technol.* **2016**, *21*, 334–343. [CrossRef]
25. Sun, K.; Wu, J.; Sun, Z.; Cao, Z. Optimal path planning method of marine sailboat based on fuzzy neural network. *J. Coast. Res.* **2019**, *93*, 911–916. [CrossRef]
26. Shen, Z.; Ding, W.; Liu, Y.; Yu, H. Path planning optimization for unmanned sailboat in complex marine environment. *Ocean Eng.* **2023**, *269*, 113475. [CrossRef]
27. Zhang, Y.; Shi, G.; Liu, J. Dynamic energy-efficient path planning of unmanned surface vehicle under time-varying current and wind. *J. Mar. Sci. Eng.* **2022**, *10*, 759. [CrossRef]
28. Ding, X.; Bian, H.; Ma, H.; Wang, R. Ship trajectory generator under the interference of wind, current and waves. *Sensors* **2022**, *22*, 9395. [CrossRef] [PubMed]
29. Precioso, D.; Milson, R.; Bu, L.; Menchions, Y.; Gómez-Ullate, D. Hybrid search method for Zermelo's navigation problem. *Comput. Appl. Math.* **2024**, *43*, 250. [CrossRef]
30. Li, S.; Li, C. Path planning for unmanned surface vehicles in dynamic environments based on artificial potential field and global guided reinforcement learning. *J. Mar. Sci. Appl.* **2026**, *25*, 575–586. [CrossRef]
31. Li, C.; Yao, L.; Mi, C. Fusion algorithm based on improved A* and DWA for USV path planning. *J. Mar. Sci. Appl.* **2025**, *24*, 224–237. [CrossRef]
32. Tan, J.; Sun, Y.; Cai, K.; Zhang, Y.; Zhang, L. Energy-Efficient AUV Path Planning Under Multiple Constraints Using a Synergistic Differential Evolution Algorithm. *J. Mar. Sci. Appl.* **2025**, 1–19. [CrossRef]
33. Wu, Y.; Wang, T.; Liu, S. A review of path planning methods for marine autonomous surface vehicles. *J. Mar. Sci. Eng.* **2024**, *12*, 833. [CrossRef]
34. González-Prieto, J.A.; Villaverde, A.F. Smooth Nonlinear High-Gain Observers for a Class of Dynamical Systems. *IEEE Access* **2021**, *10*, 3693–3704. [CrossRef]
35. González-Prieto, J.A. Practical fixed-time non-singular sliding mode control of second order nonlinear dynamic systems with chattering and overshooting avoidance. *Eur. J. Control* **2024**, *80*, 101114. [CrossRef]
36. Pirnay, H.; López-Negrete, R.; Biegler, L.T. Optimal sensitivity based on IPOPT. *Math. Program. Comput.* **2012**, *4*, 307–331. [CrossRef]

37. Blank, J.; Deb, K. Pymoo: Multi-objective optimization in python. *IEEE Access* **2020**, *8*, 89497–89509. [CrossRef]
38. Hart, W.E.; Laird, C.D.; Watson, J.-P.; Woodruff, D.L.; Hackebeil, G.A.; Nicholson, B.L.; Siirola, J.D.; Watson, J.-P.; Woodruff, D.L. *Pyomo-Optimization Modeling in Python*; Springer: Berlin/Heidelberg, Germany, 2017; Volume 67.

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