

FLUMINENSE FEDERAL UNIVERSITY

ANDRÉ PAULO DANTAS DE ARAÚJO

**Path Planning Optimization with Obstacle Avoidance for
Autonomous Sailboats Using Hierarchical Reinforcement
Learning with Potential Fields.**

NITERÓI

2026

ANDRÉ PAULO DANTAS DE ARAÚJO

Path Planning Optimization with Obstacle Avoidance for Autonomous Sailboats Using Hierarchical Reinforcement Learning with Potential Fields.

Doctoral Thesis presented to the Graduate Program in Computer of Fluminense Federal University as a partial requirement for obtaining the degree of Doctor in Computer. Area of concentration: Computer Science.

Advisor:

Esteban Walter Gonzalez Clua

Co-advisor:

Luiz Marcos Garcia Gonçalves

NITERÓI

2026

Ficha catalográfica automática - SDC/BEE
Gerada com informações fornecidas pelo autor

A658p Araujo, ANDRE PAULO DANTAS DE
Path Planning Optimization with Obstacle Avoidance for
Autonomous Sailboats Using Hierarchical Reinforcement Learning
with Potential Fields / ANDRE PAULO DANTAS DE Araujo. - 2025.
107 p.

Orientador: ESTEBAN WALTER GONZALEZ CLUA.
Coorientador: LUIZ MARCOS GARCIA GONÇALVES.
Tese (doutorado)-Universidade Federal Fluminense, Instituto
de Computação, Niterói, 2025.

1. Autonomous Sailboat. 2. Hierarchical Reinforcement
Learning. 3. Mapless Navigation. 4. Collision Avoidance. 5.
Produção intelectual. I. CLUA, ESTEBAN WALTER GONZALEZ,
orientador. II. GONÇALVES, LUIZ MARCOS GARCIA, coorientador.
III. Universidade Federal Fluminense. Instituto de
Computação. IV. Título.

CDD - XXX

ANDRÉ PAULO DANTAS DE ARAÚJO

Doctoral Thesis presented to the Graduate Program in Computer of Fluminense Federal University as a partial requirement for obtaining the degree of Doctor in Computer. Area of concentration: Computer Science.

Approved in August 2026.

EXAMINING COMMITTEE

Prof. Dr. ESTEBAN WALTER GONZALEZ CLUA - Advisor, UFF

Prof. Dr. LUIZ MARCOS GARCIA GONÇALVES - Co-advisor, UFRN

Prof. Dr. FLÁVIA CRISTINA BERNARDINI, UFF

Dr. COSIMO DISTANTE, ISASI-CNR

Dr. LUIS MARTÍ, INRIA-CHILE

Prof. Dr. RAPHAEL PEREIRA DE OLIVEIRA GUERRA, UFF

Niterói

2026

I dedicate this work to the women of my life: my daughter, wife, mother, grandmother, and sister.

“What man is a man who doesn’t make his world better.”

Acknowledgments

This work would not have been possible without the support and guidance of many individuals and institutions.

First and foremost, I would like to express my deepest gratitude to my family, whose unwavering support and encouragement have been my greatest strength throughout this journey. Your belief in me has been the foundation of all my achievements.

I extend my heartfelt thanks to Fluminense Federal University for providing me with an excellent educational environment and the resources necessary for my research. The faculty and staff have always been supportive and inspiring.

I want to give a special thanks to the INRIA Institute in France, where I had the privilege of collaborating with some of the brightest minds in the field. The experiences and knowledge I gained there have been invaluable to my work.

I am also deeply grateful to the Institute of Applied Sciences and Intelligent Systems - ISASI CNR in Italy. The collaborative opportunities and the advanced research facilities provided by ISASI CNR significantly contributed to the progress of my research.

I owe a great debt of gratitude to my advisors, whose expertise, patience, and guidance have been indispensable. Their insightful feedback and unwavering support have helped shape this work into its final form. Thank you for believing in my potential and constantly pushing me to achieve excellence.

Finally, I thank all my friends and colleagues who have been with me throughout this journey. Your support and camaraderie have made this experience truly memorable.

Resumo

O desenvolvimento de veículos marinhos autônomos exige a simulação rigorosa de seus processos de aprendizagem e controle antes da implantação no mundo real. Esta tese aborda o desafio crítico do planejamento de rotas e desvio de obstáculos para veleiros autônomos. Algoritmos de navegação tradicionais, como A^* e Otimização por Colônia de Formigas (ACO), operam sob um paradigma rígido de "planejar-para-executar", dependendo fortemente de mapas de obstáculos completos e *a priori*. Esta estrita dependência de mapas os torna estruturalmente vulneráveis em ambientes de águas abertas, onde obstáculos não mapeados e dinâmicos são onipresentes. Para superar essa limitação, esta pesquisa propõe a arquitetura de Controle por Reforço Hierárquico (HR-C). O método emprega uma estrutura de dois níveis: a orientação global de rota combina o algoritmo de busca A^* com Campos Potenciais Artificiais (APF), enquanto o controle reativo local é gerenciado por uma política aprendida através da Otimização de Políticas Proximais (PPO). Fundamentalmente, o HR-C opera sem qualquer mapa prévio de obstáculos, dependendo exclusivamente de um espaço de observação de 12 dimensões alimentado por dados LiDAR em tempo real, contabilizando explicitamente as correntes oceânicas e o vento aparente. A abordagem é avaliada comparativamente contra métodos clássicos dependentes de mapa (A^* +PID e ACO+PID) no ambiente de simulação de alta fidelidade Yara/eSailor. A avaliação abrange cinco cenários complementares: (A) navegação padrão com mapa conhecido, (B) análise de estresse de atuadores utilizando apenas as velas, (C) teste empírico de falsificação com obstáculo desconhecido, (D) ambiente dinâmico multiembarcação com seis embarcações SDF físicas como obstáculos em tempo real, e (E) uma corrida competitiva com seis agentes avaliando estratégias explícitas de acionamento de vela. Os resultados demonstram que, enquanto os métodos clássicos falham estruturalmente quando suas premissas de mapa são violadas, o $A^*/APF+PPO$ navega com segurança por ameaças não mapeadas e dinâmicas. Além do $A^*/APF+PPO$ de referência, esta tese investiga o desacoplamento do atuador de vela da política PPO. Dois módulos explícitos de controle de vela — um sistema fuzzy Mamdani e um regressor Polinomial ajustado ao diagrama polar do veleiro — são introduzidos e avaliados em uma corrida competitiva com seis agentes. Em uma corrida individual representativa, a configuração resultante $A^*/APF+PPO+Polynomial$ completou a missão em 57 passos, com distância final de 4,2 m, redução de 72% no deslocamento total do leme em relação ao PPO puro (1.106° vs. 4.003°) e zero colisões. A validação estatística com $N = 9$ execuções independentes

confirma a redução de desgaste do leme como reproduzível (redução de 51% no RMS em relação ao baseline Implícito, $p < 0,001$, Mann-Whitney U) e a segurança contra colisões como estatisticamente robusta; a velocidade de conclusão entre execuções não apresentou diferença significativa entre as variantes A*/APF+PPO ($p = 0,33$, Kruskal-Wallis), portanto a vantagem de velocidade observada numa única corrida não deve ser lida como propriedade geral da configuração Polinomial. Todos os métodos clássicos falham na corrida com obstáculos dinâmicos (A*+PID: 48 colisões, ACO+PID: 15 colisões). Esta tese apresenta evidências de que a integração do planejamento estratégico com o aprendizado por reforço guiado por sensores é necessária para a autonomia marítima em ambientes dinâmicos e não mapeados, e que o acionamento explícito da vela melhora de forma reproduzível a longevidade do hardware sem custo mensurável à velocidade da missão.

Palavras-chave: Veleiro Autônomo, Veículo de Superfície Não Tripulado, Controle por Reforço Hierárquico, Otimização de Políticas Proximais, Campos Potenciais Artificiais, Navegação sem Mapa, Prevenção de Colisões, Controle Fuzzy de Vela, Controlador Polinomial de Vela, Benchmark Multiagente.

Abstract

The development of autonomous marine vehicles requires rigorous simulation of their learning and control processes prior to real-world deployment. This thesis addresses the critical challenge of path planning and obstacle avoidance for autonomous sailboats. Traditional navigation algorithms, such as A* and Ant Colony Optimization (ACO), operate under a rigid "plan-then-execute" paradigm, heavily relying on complete, a priori obstacle maps. This strict map dependency renders them structurally vulnerable in open-water environments where uncharted and dynamic obstacles are ubiquitous. To overcome this limitation, this research proposes the Hierarchical Reinforcement Control (HR-C) framework. The method employs a two-level architecture: global path guidance combines the A* search algorithm with Artificial Potential Fields (APF), while local reactive control is managed by a policy learned through Proximal Policy Optimization (PPO). Critically, HR-C operates without any prior obstacle map, relying exclusively on a 12-dimensional observation space driven by real-time 2D LiDAR data, explicitly accounting for ocean currents and apparent wind. The approach is comparatively evaluated against classical map-dependent baselines (A*+PID and ACO+PID) in the high-fidelity Yara/eSailor simulation environment. The evaluation covers five complementary scenarios: (A) standard navigation with a known map, (B) a sail-only actuator stress analysis, (C) an empirical falsification test using a controlled unknown obstacle, (D) a dynamic multi-vessel environment with six SDF-physical vessel bots as real-time obstacles, and (E) a six-agent competitive race benchmark evaluating explicit sail actuation strategies. Results across all five scenarios demonstrate that while classical methods fail structurally or behave erratically when their map assumptions are violated, A*/APF+PPO safely navigates uncharted and dynamic threats autonomously. Beyond the baseline A*/APF+PPO, this thesis further investigates the decoupling of the sail actuator from the PPO policy. Two explicit sail modules — a Mamdani Fuzzy inference system and a Polynomial regressor fitted to the vessel's polar diagram — are introduced and benchmarked in a novel six-agent competitive race. In a single representative race run, the resulting A*/APF+PPO+Polynomial configuration completed the mission in 57 steps, at a final distance of 4.2 m, with a 72% reduction in total rudder displacement compared to the pure end-to-end PPO baseline (1,106° vs. 4,003°) and zero collisions. Statistical validation across $N = 9$ independent runs confirms the rudder-wear reduction as reproducible (51% RMS reduction versus the Implicit baseline, $p < 0.001$, Mann-Whitney U) and collision safety as statistically robust; completion speed across

runs was not significantly different between A*/APF+PPO variants ($p = 0.33$, Kruskal-Wallis), so the single-run speed advantage should not be read as a general property of the Polynomial configuration. All classical baselines fail in the dynamic-obstacle race (A*+PID: 48 collisions, ACO+PID: 15 collisions). This thesis provides evidence that integrating strategic planning with sensor-driven reinforcement learning is necessary for maritime autonomy in unmapped, dynamic environments, and that explicit sail actuation reproducibly improves hardware longevity without a measured cost to mission speed.

Keywords: Autonomous Sailboat, Unmanned Surface Vehicle, Hierarchical Reinforcement Control, Proximal Policy Optimization, Artificial Potential Fields, Mapless Navigation, Collision Avoidance, Fuzzy Sail Control, Polynomial Sail Controller, Multi-Agent Benchmark.

List of Figures

1	eBoat digital twin (F-Boat) — complete 3D reconstruction from URDF/STL meshes, shown in three views: isometric, starboard, and stern. Key structural elements are color-coded: hull/bow (blue), deck (beige), mast/boom (brown), sail (white, 3.19 m^2), rudder (red, 0.101 m^2), keel (dark gray, 0.238 m^2). Actuated joints (boom $\pm 90^\circ$, rudder $\pm 60^\circ$) are labeled in their respective panels.	25
2	eBoat physics model. (A) Sail aerodynamics: true wind V_W , vessel surge V_s , and apparent wind V_{app} combine to produce lift F_L and drag F_D as functions of the boom angle δ_b ; the forward thrust F_{thrust} is the net longitudinal component. (B) Rudder and keel hydrodynamics: the rudder (red, $A_r = 0.101 \text{ m}^2$) generates a lateral steering force; the keel (dark gray, $A_k = 0.238 \text{ m}^2$) provides the counteracting sway resistance. (C) eBoat polar diagram: contour map of surge speed as a function of boom angle and True Wind Angle (TWA); red dots mark the per-TWA optimum; the white dashed curve is the degree-4 polynomial regressor used by the A*/APF+PPO+Polynomial controller. . . .	29
3	eBoat polar diagram generated from open-loop Gazebo sweeps. Each curve plots surge velocity vs. boom angle at a fixed True Wind Angle. Green dots mark the surge-maximizing boom angle at each TWA; the green curve is the polynomial regressor fitted to these optima (see Section 2.2.3 regarding the fitted degree).	30
4	Hierarchical Reinforcement Control (HR-C) process flow diagram, illustrating the interaction between the High-Level (Strategy: A*/APF) and Low-Level (Tactics & Control: PPO) components.	43
5	Hierarchical Reinforcement Control Architecture Block Diagram, detailing the data flow from sensors to actuator outputs.	44
6	Conceptual illustration of the individual forces acting on the HR-C agent: attractive force (blue), scaled repulsive force (red, $0.1 \times F_{rep}$), ocean current force (green), and the resulting net resultant vector (black).	48

7	Grouped bar chart of key performance metrics for the standard scenario. A*+PID dominates on speed and efficiency only because it uniquely relies on a fully known environment.	60
8	XY trajectories of the three navigation methods under 9-knot tailwind conditions with static obstacles. The dashed grey line represents the theoretical optimal straight line.	61
9	Distance to the target waypoint as a function of elapsed simulation steps. A*/APF+PPO's non-monotonic profile is the mathematical signature of real-time reactive obstacle avoidance.	62
10	Distance to the nearest obstacle at each simulation step. A*/APF+PPO recovers from proximity events dynamically through APF repulsion.	63
11	Violin plots of absolute boom and rudder angle distributions. The greater variance in A*/APF+PPO demonstrates active, sensor-driven corrections in response to LiDAR readings.	63
12	Apparent wind angle histograms demonstrating A*/APF+PPO's active aerodynamic sailing.	64
13	Radar chart comparing methods across normalized criteria under known-environment conditions.	65
14	Complete rudder wear analysis across all race competitors. (A) Raw rudder angle time series. (B) Cumulative angular displacement $\Sigma \Delta\delta_r $ over steps — the slope is the instantaneous wear rate. (C) Total direction reversals N_{rev} (mechanical load cycles). (D) RMS rudder increment per step (policy aggressiveness). (E) Rudder angle distribution (violin plots). (F) Normalized wear/performance comparison across all metrics; lower is better. Gold border indicates minimum in each panel. The A*/APF+PPO+Polynomial achieves the best RMS and total displacement with zero collisions.	66

15	Complete sail (boom) actuator wear analysis. (A) Boom angle time series revealing the qualitative signature of each sail policy: the A*/APF+PPO+Polynomial produces a smooth, low-oscillation trace; the Implicit PPO scatters erratically. (B) Cumulative boom displacement $\Sigma \Delta\delta_b $ over steps. (C) Total boom displacement (91% reduction from Implicit to Polynomial). (D) Boom angle vs. True Wind Angle: the Polynomial controller traces a deterministic curve; the Fuzzy controller forms discrete horizontal bands; the Implicit PPO scatters broadly. (E) Rudder–boom cross-correlation: peak near zero for A*/APF+PPO+Polynomial confirms decoupling; high correlation for A*/APF+PPO confirms that erratic boom commands propagate as rudder jitter. (F) Normalized multi-metric summary.	67
16	Actuator effort metrics (sail-only). A*/APF+PPO actively commands actuators to navigate dynamically, whereas classical methods rely on pre-computed static pathways. . .	68
17	XY Trajectories isolating sailing capabilities by disabling auxiliary motor assistance. . .	68
18	Performance under unmapped conditions. (Left) Trajectories exposing the structural failure of A*+PID and the erratic recovery of ACO+PID. (Right) Monotonic convergence demonstrated solely by A*/APF+PPO.	70
19	Trajectories in Scenario D. A*/APF+PPO smoothly navigates around the dynamic bots; classical planners cannot update their routes in response to moving vessels.	71
20	Quantitative comparison in Scenario D: bot collisions, mission steps, and final distance to goal.	72
21	Race trajectories. The three A*/APF+PPO variants reach the goal; all classical baselines stall or overshoot due to structural incompatibility with dynamic obstacles.	73
22	Rudder actuator effort across all six race competitors	74
23	Rudder and boom angle traces over simulation steps for the three A*/APF+PPO variants. The Polynomial controller yields the smoothest rudder signal and the most stable boom trim.	74
24	Commanded boom angle vs. absolute True Wind Angle (TWA) for the three A*/APF+PPO variants during the competitive race. Each point represents one control step. The Polynomial controller traces a smooth, deterministic polar curve; the Fuzzy controller forms discrete horizontal bands (rule activations); the Implicit PPO scatters broadly, indicating inefficient aerodynamic learning.	76

- 25 Surge velocity (top) and commanded boom angle (bottom) over simulation steps for all race competitors. A*/APF+PPO variants reach cruise speed faster and maintain higher steady-state velocity. The Polynomial boom trace is the smoothest; the Implicit boom is highly oscillatory. 77
- 26 Ablation comparison across the three A*/APF+PPO architectures: Implicit (pure PPO, end-to-end), Fuzzy (Mamdani FIS boom), and Polynomial (polar-diagram regressor boom). All non-boom components are identical, isolating the sail module's effect. . . . 78
- 27 Direct pairwise comparison between A*/APF+PPO and A*/APF+PPO+Polynomial: trajectory, rudder angle trace, boom angle trace, and distance-to-goal convergence profile. 79
- 28 Pareto trade-off plane: mission steps (speed) vs. composite actuation energy. The A*/APF+PPO+Polynomial achieves the global optimum—fewest steps *and* lowest energy—breaking the traditional speed–energy trade-off. Classical baselines are dominated on both axes. 80
- 29 Multi-run statistical distributions ($N = 9$ independent races per algorithm). Each box shows median and IQR; whiskers extend to $1.5 \times \text{IQR}$; individual run points overlaid as jitter. Gold border = best mean. Significance bars compare each algorithm against the A*/APF+PPO+Polynomial winner using the Mann-Whitney U two-sided test (* $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$). The A*/APF+PPO+Polynomial achieves the lowest rudder RMS ($25.9 \pm 6.4^\circ/\text{step}$, significantly lower than all other A*/APF+PPO variants at $p < 0.001$) and near-zero collisions, confirming its dominance is statistically robust across runs. 82
- 30 Mann-Whitney U pairwise p -value heatmap 83
- 31 Mean $\pm 95\%$ CI (Student- t , $N = 9$) for all six metrics. The A*/APF+PPO+Polynomial is the *only* algorithm achieving simultaneously low rudder RMS ($25.9 \pm 6.4^\circ/\text{step}$) *and* near-zero collisions (0.2 ± 0.7) *and* competitive completion speed (59.8 ± 32.0 steps). A*/APF+PPO+Fuzzy is faster on average (45.4 ± 15.8 steps) but pays a $2.6\times$ rudder wear penalty. Classical planners occupy the high-collision, moderate-speed region. . . . 84

32 Consolidated statistical summary (mean $\pm$ std, $N = 9$). Superscripts: * $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$ vs A*/APF+PPO+Polynomial (Mann-Whitney U). Green row = A*/APF+PPO+Polynomial. The table establishes that the A*/APF+PPO+Polynomial’s superior rudder efficiency is statistically confirmed, while its speed advantage over other A*/APF+PPO variants is directionally positive but not yet statistically significant at this sample size. 85

List of Tables

1	High-Level Evaluation of Compared Navigation Approaches	23
2	Key physical parameters of the eBoat URDF model, as configured in the Gazebo plugins.	26
3	Identified hydrodynamic drag coefficients as specified in the URDF/Gazebo plugin. . .	27
4	eBoat onboard sensor specifications as configured in the URDF/Gazebo model.	31
5	Summary of Simulators Used in Related Literature	32
6	Representative Obstacle Avoidance Techniques Categorized by Approach	35
7	High-level $\times$ low-level $\times$ sail-layer combinations	40
8	12-Dimensional Observation Space of the A*/APF+PPO Agent. Features r_1 – r_2 encode the APF resultant vector; r_{10} encodes LiDAR proximity. Feature r_6 is removed for the Fuzzy and Polynomial sail-layer configurations (11-dim).	46
9	PPO Hyperparameters used for all A*/APF+PPO variants. Configured via the <code>stable-baselines3</code> library.	50
10	Four-phase curriculum training schedule. Wind speed and angle are randomized within the shown ranges at each episode reset.	50
11	Representative subset of the 15-rule Mamdani fuzzy rule base. Each cell shows the output boom trim. Blank cells are “don’t-care” combinations.	52
12	Algorithms evaluated in each scenario	58
13	Quantitative Performance Summary, Standard Scenario	60
14	Actuator effort metrics isolating mechanical stress (sail-only, with obstacles).	66
15	Unknown obstacle experiment results. A*/APF+PPO achieves the only operationally viable success when the map assumption is violated.	69

16	Dynamic Bots scenario results (Scenario D). Bot collisions occur when the planner’s map cannot represent moving vessels.	71
17	Competitive race results (dynamic-obstacle scenario). Best value per column in bold . † did not reach goal.	73
18	Complete actuator effort breakdown for all race competitors. $\Sigma \Delta $ = total angular displacement; N_{rev} = direction reversals; RMS = root-mean-square of incremental displacements per step. Best value per column in bold (among agents that reached the goal).	81
19	Consolidated five-scenario evaluation summary	86

Contents

1	Introduction	12
1.1	The Map Dependency Problem	12
1.2	Theme, Research Question and Problem	13
1.3	Research Hypothesis and Contribution	14
	Publications Originating from this Thesis	16
1.4	Text Structure	17
2	Background	18
2.1	Path Planning	18
2.1.1	Graph-Search-Based Algorithms: A*	18
	Why A*, and not an incremental or dynamic planner such as D*?	19
2.1.2	Metaheuristic Optimization: Ant Colony Optimization	19
	Why ACO, and not Genetic Algorithms or Particle Swarm Optimization?	21
2.1.3	Reactive Navigation: Artificial Potential Fields	21
2.1.4	Reinforcement Learning: PPO and Hierarchical Control	21
	Terminological note: agent vs. multi-vessel vs. multi-agent.	23
2.2	Simulation Environment: The eSailor Ecosystem	24
2.2.1	eBoat Digital Twin and Actuator Modeling	25
2.2.2	Hydrodynamic Model	26
2.2.3	Sail Aerodynamics and Rudder Hydrodynamics	27
2.2.4	Onboard Sensor Suite	31

2.3	Other Simulation Environments for Maritime Robotics	31
3	Related Works	33
3.1	Classical and Metaheuristic Path Planning	33
3.2	Reinforcement Learning and Deep Learning for Maritime Navigation	34
3.3	Obstacle Avoidance Techniques	34
3.4	Autonomous Sailboats and Energy-Efficient Maritime Systems	36
3.5	Strictly Related Works	36
4	Architectural Design and Implementation of the HR-C Framework	38
4.1	Hierarchical Layer Taxonomy	38
	High-level (strategic) layer.	38
	Low-level (tactical) layer.	39
	Sail-actuation layer (optional third layer).	39
4.2	Comparative Algorithm Implementation	39
4.2.1	A*+PID (Non-Reactive Baseline)	40
	Grid parameters.	40
	PID heading controller.	40
	A*(RP)+PID variant.	41
4.2.2	ACO+PID (Reactive-Layer Baseline)	41
	ACO parameters.	42
	PID reactive layer.	42
4.2.3	A*/APF+PPO (Proposed Mapless Architecture)	42
4.3	The Hierarchical Architecture	44
4.4	12-Dimensional Observation Space Engineering	44
4.5	Feature Engineering: APF as Guidance	46

4.6	Integration of Ocean Currents	46
4.7	Reward Function Formalization	47
4.8	Curriculum Learning and Robustness Training	49
4.9	Explicit Sail Actuation Modules	50
4.9.1	Fuzzy Sail Controller (A*/APF+PPO+Fuzzy)	52
4.9.2	Polynomial Sail Controller (A*/APF+PPO+Polynomial)	52
4.9.3	Integration and Architectural Impact	53
4.10	Computational Complexity Analysis	53
4.10.1	Classical Baselines: A* and ACO	55
4.10.2	A*/APF+PPO: Training and Runtime Cost	55
	Runtime (deployment) cost.	55
	Training cost.	56
5	Experiments and Results	57
5.1	Experimental Setup and Evaluation Scenarios	57
5.1.1	Scenario A: Standard Navigation (Baseline Comparison)	58
5.1.2	Scenario B: Sail-Only Actuator Stress Analysis	58
5.1.3	Scenario C: Unknown-Obstacle Robustness (Falsification Test)	59
5.1.4	Scenario D: Dynamic Multi-Vessel Environment (Moving Obstacles)	59
5.1.5	Scenario E: Explicit Sail Actuation Benchmark (Competitive Race)	59
5.2	Performance Under Standard Conditions (Idealized Map)	59
5.2.1	Trajectory and Convergence Analysis	61
5.2.2	Obstacle Clearance and Control Effort	62
5.2.3	Wind Utilization and Overall Performance (Scenario A)	63
5.3	Actuator Mechanical Stress (Sail-Only Analysis)	65
5.4	Empirical Falsification: Unknown Obstacle Robustness	68

A*+PID (Failure):	69
ACO+PID (Erratic Success):	69
A*/APF+PPO (Smooth Success):	69
5.5 Dynamic Multi-Vessel Scenario (Scenario D)	70
5.6 Competitive Race: Explicit Sail Actuation Benchmark	72
Sail Ablation.	75
Classical Baseline Failure.	75
5.6.1 Sail Policy Behavior: Boom Angle vs. True Wind Angle (Scenario E)	75
5.6.2 Speed and Actuator Time-Series Analysis (Scenario E)	77
5.6.3 Speed–Energy Pareto Analysis (Scenario E)	79
5.7 Statistical Robustness Analysis	81
5.7.1 Multi-Run Boxplot Analysis	81
5.7.2 Global and Pairwise Statistical Tests	82
5.7.3 Mean $\pm$ CI ₉₅ and Consolidated Summary	83
5.8 Consolidated Cross-Scenario Summary	86
6 Conclusion	88
6.1 Limitations	89
6.2 Future Work	90
REFERENCES	92

1 Introduction

The increasing global demand for water resources coupled with the escalating impacts of climate change has made the continuous monitoring of aquatic ecosystems a priority for environmental agencies. According to the State Institute for the Environment of Rio de Janeiro (INEA-RJ), persistent monitoring is the only viable path for effective environmental licensing and public health management. However, traditional monitoring via manned vessels is limited by high operational costs and limited spatial and temporal resolution.

The F-Boat project is a collaborative initiative between Fluminense Federal University (UFF) and the Federal University of Rio Grande do Norte (UFRN) that addresses these limitations by developing an autonomous sailboat for persistent water quality monitoring. Wind propulsion makes these vessels a sustainable, high-endurance platform for environmental sensing. The autonomous navigation of a sailboat presents a unique challenge: the agent must co-optimize sail trim and heading while respecting strict aerodynamic constraints and reacting to dynamic environmental disturbances, such as rain, wind gusts, currents, and unexpected or dynamic obstacles.

Although partial solutions have recently been developed addressing some of these issues, navigation among dynamic obstacles remains unsolved. Traditional algorithms do not fit sailboats well because they do not behave in the same way as terrestrial robots (S. JÚNIOR et al., 2013; SANTOS et al., 2016).

1.1 The Map Dependency Problem

Traditionally, in the field of maritime robotics, trajectory planning has historically relied on a "plan-reason-execute" paradigm. Algorithms such as A* and Ant Colony Optimization (ACO) have been widely adopted. However, they suffer from a fundamental flaw, which is the *map dependency*. In general, these methods require a complete, static, and accurate map of the environment before departure.

In real-world marine operations, obtaining such a reliable map is impossible due to dynamic ele-

ments. Floating debris, moving vessels, and unmapped coastal changes are inherent to the environment. When a map-dependent planner encounters an unmapped obstacle, it lacks the structural logic to reason and adapt its trajectory, leading to catastrophic mission failure. Recent literature underscores that even small map inaccuracies can invalidate entire planned segments, driving a paradigm shift toward sensor-driven mapless autonomy normally using exteroceptive sensors like 2D LiDAR (SELLERS et al., 2026). In this thesis, we argue that a fundamental paradigm shift from map-based planning to sensor-driven, mapless autonomy is required for the operations of autonomous sailboats in open waters, aligning with the 2025-2026 consensus that favors hybridizing topological guidance with reactive Deep Reinforcement Learning (DRL) (ZHANG, H. et al., 2025).

1.2 Theme, Research Question and Problem

The theme of this thesis can be roughly enunciated as the study and development of a robust navigation algorithm for sailboats operating in dynamic, unknown environments based on a hybrid paradigm involving Deep Hierarchical Reinforcement Learning and Artificial Potential Field.

Inside this theme and having in mind the Map Dependency Problem, this doctoral research is driven by five primary questions that seek to define how sailboats navigate in unknown environments:

1. **RQ1:** How can a hierarchical architecture combining a classical strategic layer with a learned tactical layer be designed to eliminate map dependency while ensuring global waypoint convergence?
2. **RQ2:** Can Artificial Potential Fields (APF) be integrated into a Deep Reinforcement Learning observation space to provide reactive safety without overriding the learned aerodynamic control policy?
3. **RQ3:** What is the quantifiable mechanical trade-off between the safety of reactive, mapless autonomy and the actuator stress compared to pre-planned optimal routes?
4. **RQ4:** Does such a mapless, sensor-driven architecture generalize to dynamic multi-vessel environments where other vessels act as moving physical obstacles undetectable by static planners?
5. **RQ5:** Can the sail actuation responsibility be structurally decoupled from the PPO network and replaced by an explicit model-based module, such as a Fuzzy inference system or a Polynomial polar-diagram regressor, to simultaneously improve mission speed and reduce actuator wear?

These questions can be joined together in a single question that consists of the main research problem which will be worked on this thesis, as described next.

Problem:

Is it feasible, and how safe, robust, and efficient can a navigation algorithm based on a hybrid paradigm involving Deep Hierarchical Reinforcement Learning and Artificial Potential Field be for sailboats operating in unknown, dynamic environments?

1.3 Research Hypothesis and Contribution

Hence, addressing the aforementioned research problem leads to the formulation of the central research hypothesis of this thesis, stated below, followed by the corresponding research contributions developed to validate it.

Hypothesis:

The main hypothesis the thesis intends to argue is that it is possible to plan and execute trajectory paths that allow a safe, robust, and efficient navigation of a sailboat operating in unknown, complex, and dynamic environment with reduction of mechanical stress by way of using a hybrid algorithm involving Deep Hierarchical Reinforcement Learning and Artificial Potential Field.

The verification of this hypothesis goes throughout the implementation and empirical experiments and tests, in order to verify the efficiency of the proposed technique when facing the best known algorithms. Also, mechanical stress of the sailboat is addressed, as well as dynamic situations in which obstacle avoidance is necessary.

The verification of the above hypothesis provides the main contribution of this work, as detailed in sequence.

Contribution:

Development of a new hybrid navigation strategy using Deep Hierarchical Reinforcement Learning and Artificial Potential Field, which can be used by USV for autonomous navigation.

To reach this main contribution, we provide five (original) secondary contributions (new techniques or steps necessary during the verification of the above hypothesis) to the fields of Autonomous Surface Vehicles (USVs) and Artificial Intelligence. Some of them are new techniques necessary in the steps followed to reach the main contribution:

1. **The Hierarchical Reinforcement Control (HR-C) + APF Framework:** A novel hierarchical navigation architecture that combines high-level A* strategic reference with Artificial Potential Field (APF) reactive guidance and a low-level Proximal Policy Optimization (PPO) tactical agent driven by 2D LiDAR, enabling fully mapless, sensor-driven navigation.
2. **The Unknown-Obstacle Falsification Test:** A rigorous experimental demonstration of the structural failure of map-dependent algorithms (A* and ACO) when confronted with unmapped threats. Adversarial falsification testing has recently emerged as a critical standard for evaluating rule compliance and safety in maritime RL agents before real-world deployment (MÜLLER et al., 2025). This test directly proves the necessity of the proposed HR-C+APF model.
3. **Mechanical Stress Analysis of Mapless Sailing:** A pioneering evaluation of actuator wear (rudder reversals and angular displacement) in reactive sailboats. Given that high-frequency control signals generated by DRL reactive loops are now recognized by the 2025-2026 literature as major causes of mechanical stress and actuator fatigue (KRAUTWIG et al., 2026), this analysis establishes the "price of autonomy" for future hardware design and highlights the critical need for control signal quality evaluation (AGYEI; SARHADI; POLANI, 2025).
4. **Dynamic Multi-Vessel Obstacle Scenario:** An evaluation of HR-C in a scenario where six SDF-physical vessels, following scripted trajectories, patrol the navigation corridor as moving obstacles undetectable by any static map. The experiment confirms that HR-C+APF's real-time LiDAR-driven policy navigates this scenario successfully, while classical planners (A* and ACO) remain structurally blind to dynamically placed vessels.
5. **Explicit Sail Actuation Modules and Competitive Race Benchmark:** The structural decoupling of the sail actuator from the PPO policy, enabling two plug-in modules—a Mamdani Fuzzy inference system and a Polynomial polar-diagram regressor—to override the boom command. A novel six-agent competitive race, with six dynamic physical vessel obstacles, serves as the evaluation arena. In a single representative race run, the resulting A*/APF+PPO+Polynomial configuration reaches the goal in 57 steps, at a 4.2 m final distance, with a 72% reduction in total rudder displacement versus the end-to-end PPO baseline; across $N = 9$ independent runs (Section 5.7), this rudder-wear reduction is the statistically confirmed result, while completion speed does not differ significantly between A*/APF+PPO variants.

Publications Originating from this Thesis

The following papers were produced in direct relation to the research developed in this thesis; not all of them originated from the specific research questions answered here (Section 1.2), but all report work carried out as part of this thesis's research program. Items marked **[Under Review]** or **[Submitted]** are currently under evaluation. Earlier, more loosely related publications by the candidate are instead cited as prior work in Chapter 3 (Section 3.4).

1. **[Under Review]** Araújo, A. P. D.; Mankina, V.; Guerra, R.; Brandão, D. N.; Vasconcellos, E. C.; Pavanelli, E. B.; Distante, C.; Leo, M.; Preux, P.; Gonçalves, L. M. G.; Clua, E. W. G. "Hierarchical Reinforcement Learning for Autonomous Sailboat Navigation in Unmapped Environments." *ACM Transactions on Autonomous and Adaptive Systems*, 2026.
2. **[Submitted]** Araújo, A. P. D.; Guerra, R.; Brandão, D. N.; Vasconcellos, E. C.; Gonçalves, L. M. G.; Clua, E. W. G. "Polynomial and Fuzzy Sail Actuation within a Hierarchical Reinforcement Control Framework for Autonomous Sailboats: A Competitive Multi-Agent Benchmark." *Ocean Engineering*, 2026.
3. Vasconcellos, E. C.; Negreiros, Á. P. F.; de Araújo, A. P. D.; Guerra, R.; Preux, P.; dos Santos, D. H.; Gonçalves, L. M. G.; Clua, E. W. G. "Yara: An Ocean Virtual Environment for Research and Development of Autonomous Sailing Robots and Other Unmanned Surface Vessels." *Journal of Intelligent & Robotic Systems*, vol. 111, n. 3, 2025. ([VASCONCELLOS; NEGREIROS, et al., 2025](#)).
4. Mankina, V.; Araújo, A. P. D.; Guerra, R.; Clua, E. W. G.; Cernicchiaro, C.; Gonçalves, L. M. G.; Distante, C. "AI-Based Autonomous Sailboat Navigation: A Review." *Journal of Field Robotics*, vol. 42, 2025. ([MANKINA et al., 2025](#)).
5. Araújo, A.; Guerra, R.; Preux, P.; Gonçalves, L.; Distante, C.; Brandão, D.; Mankina, V.; Vasconcellos, C.; Clua, E. "Towards an Autonomous Sailboat Navigation Control Architecture." In: *2024 Latin American Robotics Symposium (LARS)*, IEEE, 2024. ([ARAÚJO; GUERRA, et al., 2024](#)).
6. Araújo, A.; Preux, P.; Gonçalves, L.; Cernicchiaro, C.; Distante, C.; Brandão, D.; Mankina, V.; Vasconcellos, C.; Clua, E. "A Comparison of DRL with APF and A* with PI Control for Trajectory Planning with Obstacle Avoidance for Sailboat Robots." In: *2024 Latin American Robotics Symposium (LARS)*, IEEE, 2024. ([ARAÚJO; PREUX, et al., 2024](#)).

7. Vasconcellos, E. C.; Sampaio, R. M.; Araújo, A. P. D.; Clua, E. W. G.; Preux, P.; Guerra, R.; Gonçalves, L. M. G.; Martí, L.; Lira, H.; Sanchez-Pi, N. “Reinforcement-learning Robotic Sailboats: Simulator and Preliminary Results.” *NeurIPS 2023 Workshop on Robot Learning: Pretraining, Fine-Tuning, and Generalization with Large Scale Models*, New Orleans, United States, 2023. ([VASCONCELLOS; SAMPAIO, et al., 2023](#)).
8. Araújo, A. P. D.; Chandrasekharan, G. P.; Clua, E. W. G.; Preux, P.; Vasconcellos, E. C.; Gonçalves, L. M. G. “Vision of the Seas: Open Visual Perception Framework for Autonomous Sailing Vessels.” In: *2023 30th International Conference on Systems, Signals and Image Processing (IWSSIP)*, IEEE, 2023. ([ARAÚJO, André P. D. et al., 2023](#)).

1.4 Text Structure

The remainder of this thesis is organized as follows: Chapter 2 provides the technical background, covering the core algorithms, the eSailor simulation environment, the eBoat digital twin physics, and the onboard sensor suite. Chapter 3 surveys the related literature, organized from broad categories of maritime navigation methods to the works most strictly related to this thesis. Chapter 4 presents the mathematical implementation of the HR-C+APF model, the implementation of the comparative baselines, and the explicit Fuzzy and Polynomial sail actuation modules. Chapter 5 describes the five evaluation scenarios (A–E), presents the experimental results across all scenarios, and provides the competitive race and statistical robustness analysis. Finally, Chapter 6 summarizes the primary findings, outlines the original contributions, and presents future perspectives for real-world deployment on the physical F-Boat platform.

2 Background

This chapter provides the technical foundation required to understand the contributions of this thesis. It covers the core algorithmic building blocks used in autonomous maritime navigation, the simulation infrastructure and digital-twin models employed throughout the experiments, and the sensor suite that defines the agent’s perception of its environment. The intent is to give the reader the conceptual and mathematical tools necessary to follow the architectural decisions described in Chapter 4 and the experimental analysis in Chapter 5.

2.1 Path Planning

Path planning is commonly divided into global and local planning (SANTOS et al., 2016). Global path planning focuses on generating an efficient overall route, often using graph-search methods such as A* or metaheuristics such as Ant Colony Optimization (ACO) to find paths that account for environmental constraints. Local path planning deals with immediate concerns such as dynamic obstacles, real-time weather adjustments, and sudden environmental changes, typically addressed by reactive methods such as Artificial Potential Fields (APF) or learned policies from reinforcement learning. The effective integration of these two planning levels is crucial for comprehensive navigation systems, and it is precisely this division that our proposed HR-C framework formalizes.

2.1.1 Graph-Search-Based Algorithms: A*

The A* algorithm (HART; NILSSON; RAPHAEL, 1968) is a widely used informed search algorithm for pathfinding on graphs or grids. It efficiently finds the shortest path based on a cost function $f(n) = g(n) + h(n)$, where $g(n)$ is the exact cost from the start and $h(n)$ is an admissible heuristic (typically Euclidean distance to the goal). A* expands nodes in a priority queue ordered by f , guaranteeing optimality on a finite graph when h is admissible. In this thesis, the path generated by A* is executed using PID controllers (A*+PID baseline), and its runtime adaptability is limited to the PID controller’s

reactions unless replanning is explicitly triggered (the $A^*(RP)$ variant). Algorithm 2.1 formalizes the A^* search with optional replanning used in the experiments.

Why A^* , and not an incremental or dynamic planner such as D^* ? Incremental replanning algorithms — D^* (STENTZ, 1994), D^* Lite (KOENIG; LIKHACHEV, 2002), and their descendants — are designed to reuse information from a previous search when the *known* map changes, making repeated replanning cheaper than recomputing from scratch. This thesis does not adopt them, for two reasons specific to its experimental design. First, for the classical map-dependent baselines (A^*+PID , $A^*(RP)+PID$, $ACO+PID$), the object under test is precisely the structural map-dependency paradigm, not the computational efficiency of any particular replanning strategy: D^* still requires an occupancy-grid representation of the environment to search over, so it would suffer the same falsification-test failure mode as A^* the moment an obstacle is absent from that grid (Chapter 5, Scenario C). The $A^*(RP)$ variant already grants the classical paradigm a reactive replanning capability, functionally analogous to what D^* -style incremental search would offer, without changing the underlying conclusion. Second, and more fundamentally, A^* plays a different role inside HR-C itself: its high-level layer (Table 7) runs A^* on a grid that is permanently *empty*, so there is no obstacle information to incrementally reuse in the first place — the incremental-search advantage of D^* -family algorithms is inapplicable by construction, since all obstacle reactivity in HR-C is delegated to the APF-augmented PPO layer, not to the strategic planner. A^* , being the simplest and most widely reported baseline in the path-planning literature (Chapter 3), was therefore chosen as the standard, comparable reference point for both roles.

2.1.2 Metaheuristic Optimization: Ant Colony Optimization

Randomization methods based on optimization techniques use stochastic algorithms to solve combinatorial problems where deterministic methods fail. Instead of computing exact gradients, these methods intelligently guide random sampling toward a global optimum.

Ant Colony Optimization (ACO) (DORIGO; MANIEZZO; COLORNI, 1996) is a metaheuristic inspired by the foraging behavior of real ants, in which virtual agents probabilistically construct paths guided by a pheromone matrix τ_{ij} and a heuristic attractiveness $\eta_{ij} = 1/d_{ij}$. After each iteration, the global pheromone trail is updated: edges belonging to shorter paths receive more pheromone, while all edges undergo evaporation at rate ρ . The paths found are not guaranteed to be optimal, potentially leading to longer traveled distances. In this thesis, the path found by ACO is executed via PID controllers (ACO+PID baseline). Algorithm 2.2 details the ACO procedure ($n_{\text{ants}} = 100$, $n_{\text{iter}} = 100$, $\alpha = 1$, $\beta = 2$, $\rho = 0.1$).

Algorithm 2.1 A* Path Planning with Optional Replanning (A*(RP))

Require: Start s , goal g , obstacle map M , waypoint tolerance ε_{wp}

Ensure: Sequence of waypoints $\mathcal{W}$

```

1: function ASTARPLAN( $s, g, M$ )
2:   open  $\leftarrow \{s\}$ ; closed  $\leftarrow \emptyset$ 
3:    $g_{\text{score}}[s] \leftarrow 0$ ;  $f_{\text{score}}[s] \leftarrow \|s - g\|$ 
4:   while open  $\neq \emptyset$  do
5:      $n \leftarrow \arg \min_{n' \in \text{open}} f_{\text{score}}[n']$ 
6:     if  $n = g$  then return RECONSTRUCTPATH( $n$ )
7:     end if
8:     open  $\leftarrow \text{open} \setminus \{n\}$ ; closed  $\leftarrow \text{closed} \cup \{n\}$ 
9:     for each 8-connected neighbor  $n'$  of  $n$  with  $M[n'] \neq \text{OBSTACLE}$  do
10:      if  $n' \in \text{closed}$  then continue
11:      end if
12:       $g_{\text{cand}} \leftarrow g_{\text{score}}[n] + \|n - n'\|$ 
13:      if  $g_{\text{cand}} < g_{\text{score}}[n']$  then
14:        parent[ $n'$ ]  $\leftarrow n$ ;  $g_{\text{score}}[n'] \leftarrow g_{\text{cand}}$ 
15:         $f_{\text{score}}[n'] \leftarrow g_{\text{cand}} + \|n' - g\|$ ; open  $\leftarrow \text{open} \cup \{n'\}$ 
16:      end if
17:    end for
18:  end while
19:  return FAILURE
20: end function
21:  $\mathcal{W} \leftarrow \text{ASTARPLAN}(p_0, g, M)$ ;  $k \leftarrow 0$ 
22: while goal not reached do
23:   if  $\|p - \mathcal{W}_k\| \leq \varepsilon_{wp}$  then  $k \leftarrow k + 1$ 
24:   end if
25:   if replanning mode and obstacle detected by LiDAR then
26:      $\mathcal{W} \leftarrow \text{ASTARPLAN}(p, g, M)$ ;  $k \leftarrow 0$ 
27:   end if
28:    $\delta_r, m \leftarrow \text{PID}(p, \mathcal{W}_k)$ ; apply actuators  $(\delta_r, 90^\circ, m)$ 
29: end while

```

Why ACO, and not Genetic Algorithms or Particle Swarm Optimization? Chapter 3 situates ACO alongside Genetic Algorithms (GA) and Particle Swarm Optimization (PSO) as the three metaheuristics most commonly applied to ASV path planning. ACO was selected as the representative metaheuristic baseline for two reasons. First, its pheromone-trail mechanism produces a single incumbent path that degrades gracefully under a reactive PID deflection layer (ZHENG et al., 2022), giving the fairest possible metaheuristic baseline: like $A^*(RP)+PID$, an ACO+PID stack with a proximity-triggered deflection is already preceded in the literature, whereas GA and PSO are typically deployed as one-shot offline optimizers without an established reactive-execution counterpart, which would make a GA+PID or PSO+PID baseline a non-standard construction specific to this thesis rather than a reproducible comparison point. Second, all three metaheuristics share the same structural map-dependency this thesis falsifies: GA and PSO also optimize a candidate path against a known occupancy grid and do not sense the environment at runtime, so substituting either for ACO would not change the qualitative outcome of Scenario C, only the specific (still map-dependent) trajectory produced.

2.1.3 Reactive Navigation: Artificial Potential Fields

Introduced by Khatib (1986) (KHATIB, 1986), Potential Field Methods (PFM) use virtual attractive and repulsive forces for navigation. The goal exerts an attractive potential, while obstacles exert repulsive potentials, guiding the agent along the combined gradient. While usable standalone, classic APF often suffers from local minima. In the HR-C framework, APF calculations are not used for direct control; instead, they are integrated as input features into the RL agent’s observation space, providing explicit, reactive guidance for goal direction and obstacle proximity that the learned policy exploits alongside other sensor data.

2.1.4 Reinforcement Learning: PPO and Hierarchical Control

Reinforcement Learning (RL) (SUTTON; BARTO, 2018) enables agents to learn optimal policies through environmental interaction. The Proximal Policy Optimization (PPO) algorithm is an on-policy gradient method that clips the policy update to prevent destabilizing large steps, making it robust to hyperparameter choices and effective for continuous control tasks.

Hierarchical Reinforcement Control (HR-C) (RUAN; DAI; YU, 2023; WANG; JIANG; WANG, 2023) structures the learning problem into two levels: a high-level component that manages global path strategy by generating waypoints (using A^*), and a low-level agent that focuses on local control (using PPO) to reach each waypoint while reacting to obstacles in real time. This decomposition

Algorithm 2.2 Ant Colony Optimization Path Planning (ACO + PID)

Require: Grid G , start s , goal g , n_{ants} , n_{iter} , α , β , ρ , Q
Ensure: Best path $\mathcal{W}^*$

```

1: Initialize  $\tau_{ij} \leftarrow \tau_0$  for all  $(i,j) \in G$ 
2:  $\mathcal{W}^* \leftarrow \text{NULL}$ ;  $L^* \leftarrow \infty$ 
3: for iter = 1, ...,  $n_{\text{iter}}$  do
4:   for each ant  $a = 1, \dots, n_{\text{ants}}$  do
5:      $\mathcal{W}_a \leftarrow \text{CONSTRUCTPATH}(s, g, \tau, \alpha, \beta)$ ;  $L_a \leftarrow \text{length}(\mathcal{W}_a)$ 
6:     if  $L_a < L^*$  then  $L^* \leftarrow L_a$ ;  $\mathcal{W}^* \leftarrow \mathcal{W}_a$ 
7:     end if
8:   end for
9:    $\tau_{ij} \leftarrow (1 - \rho) \tau_{ij}$  (evaporation)
10:  for each ant  $a$  do
11:    for each edge  $(i,j) \in \mathcal{W}_a$  do
12:       $\tau_{ij} \leftarrow \tau_{ij} + Q/L_a$  (deposit)
13:    end for
14:  end for
15: end for
16: return  $\mathcal{W}^*$ 
17: function CONSTRUCTPATH( $s, g, \tau, \alpha, \beta$ )
18:   path  $\leftarrow [s]$ ; current  $\leftarrow s$ 
19:   while current  $\neq g$  do
20:      $\mathcal{N} \leftarrow$  unvisited neighbors of current
21:      $P(j) \propto \tau_{\text{current},j}^\alpha \cdot \eta_{\text{current},j}^\beta$  for  $j \in \mathcal{N}$ 
22:     next  $\leftarrow$  sample  $\mathcal{N}$  with distribution  $P$ 
23:     path  $\leftarrow \text{path} \cup \{\text{next}\}$ ; current  $\leftarrow \text{next}$ 
24:   end while
25:   return path
26: end function

```

improves learning efficiency for complex, long-horizon tasks by separating strategic planning from tactical execution. Recent architectures have further validated this hierarchical approach: models using RRT paths to guide DQN agents accelerate training convergence and improve safety against irregular obstacles (ZHANG, H. et al., 2025), and frameworks that construct sparse local graphs from LiDAR point clouds handle unseen dynamic obstacles without prior maps (SELLERS et al., 2026). In the HR-C framework, a PPO agent forms the low-level controller, trained on a 12-dimensional observation that fuses navigation, wind, LiDAR, and APF data. Table 1 summarizes the high-level characteristics of the navigation paradigms compared in this thesis.

Terminological note: agent vs. multi-vessel vs. multi-agent. This thesis uses *agent* in the precise reinforcement-learning sense: an entity that perceives a state, executes a learned policy, and receives a reward, applicable to the ego-sailboat controlled by PPO. Scenario D (Section 5.1.4) introduces additional vessels that follow scripted sinusoidal and straight-line trajectories, with no perception, policy, or reaction to the ego-vessel; they are therefore *dynamic vessel obstacles*, not agents in this sense, and Scenario D is described throughout this thesis as a dynamic *multi-vessel* environment rather than a multi-agent one. Scenario E (Section 5.1.5), in contrast, is legitimately multi-agent in the classical sense of Wooldridge (WOOLDRIDGE, 2009): each of the six competing sailboats is driven by an independent, autonomous decision-making control stack (A*+PID, ACO+PID, or one of the three A*/APF+PPO configurations) that perceives and acts within a shared environment, even though the controllers do not communicate or negotiate with one another — a *non-cooperative*, concurrent multi-agent setting, not a coordinated one.

Table 1: High-Level Evaluation of Compared Navigation Approaches

Approach	Planning Time	Path Optimality	Runtime Adaptability	Map Required?
Classical / Metaheuristic (Map-Dependent)				
A*+PID	Moderate (offline)	High (on graph)	None (laser ignored)	Yes
A*(RP)+PID	Moderate (offline) + occasional replans	High (on graph)	Partial (LiDAR-triggered full replan)	Yes
ACO+PID	High (iterative)	Variable	Partial (reactive PID, no replanning)	Yes
Hierarchical Reinforcement Control (Mapless)				
A*/APF+PPO (+Cur-rents)	N/A (learned policy)	Learned	Very high (sensor-driven at every step)	No

2.2 Simulation Environment: The eSailor Ecosystem

Simulation environments provide a risk-free, cost-effective, and scalable virtual testbed to develop, test, and train navigation strategies for USV systems. Simulators are essential for validating new ideas and optimizing algorithms before real-world deployment, functioning as an essential bridge between conceptual design and physical reality. Several core motivational aspects can be enumerated for using simulation in USV (PARAVISI et al., 2019). First, and most important, simulation guarantees safety and risk elimination, with zero hardware damage to real robots, which can crash into themselves or their surroundings by programming or logic errors. Simulation allows for aggressive trial-and-error without destroying expensive physical prototypes. In addition, extreme scenario testing is allowed, such as the simulation of catastrophic failures, dangerous environments (e.g., radiation, deep-sea, space), and edge-case scenarios. It would be impossible or unsafe for developers to test navigation in these scenarios in the real world (JORGE et al., 2019).

A further motivation is accelerated Machine Learning and AI training. Simulators generate large volumes of training images, depth maps, and sensor data to bootstrap computer vision models (SALGADO; BEZERRA, et al., 2024; MENESES SANTOS et al., 2026). Scalable reinforcement learning platforms such as NVIDIA Isaac Sim can run thousands of parallel simulations in the cloud, training navigation policies in a matter of hours — a comparable approach to the one adopted in this work (SALGADO; VASCONCELLOS, et al., 2026).

Another advantage is the speed of development with consequent cost reduction by way of iterative engineering. Simulation allows engineers to rapidly tweak a robot’s dimensions, actuators, or sensor placement and instantly see the results, drastically speeding up the hardware and software design cycle. Architectures based on software-in-the-loop (SIL) testing can be developed, so software updates and control policies can be tested on digital twins before ever being uploaded to physical machines (DE NEGREIROS et al., 2023).

These advantages motivate a strong focus on bridging the gap between simulation and reality through “Sim-to-Real” pipelines. By combining physically accurate physics engines with domain randomization, AI models trained in virtual environments can be transferred to operate reliably in unstructured real-world conditions (VASCONCELLOS; SAMPAIO, et al., 2023).

In this work, all experiments and training procedures were conducted within the **Yara** simulation environment (VASCONCELLOS; NEGREIROS, et al., 2025), referred to as eSailor in this thesis. Yara is an open oceanic virtual environment built upon the Robot Operating System (ROS Noetic) and Gazebo Classic 11, specifically designed for developing, training, and evaluating autonomous agents

operating digital twins of sailing robots. Unlike simplified grid-world simulations, Yara couples the ODE (Open Dynamics Engine) with custom foil-dynamics plugins that compute real-time lift and drag forces on the sail, rudder, and keel, validated against physical tank tests of the F-Boat platform. A key practical advantage is a simulation speedup of approximately $300\times$ on consumer hardware, enabling 10^6 PPO timesteps in approximately eight hours.

2.2.1 eBoat Digital Twin and Actuator Modeling

The central agent of this research is the eBoat, a high-fidelity digital twin of the physical F-Boat robotic sailboat whose foil dynamics are constructed based on the real vessel and validated within the Yara environment (VASCONCELLOS; NEGREIROS, et al., 2025). The 3D model is defined in a URDF (Unified Robot Description Format) file consisting of 18 rigid links connected by revolute and fixed joints, exported from CAD and converted to STL/DAE meshes. Figure 1 shows the rendered vessel inside the Gazebo environment.

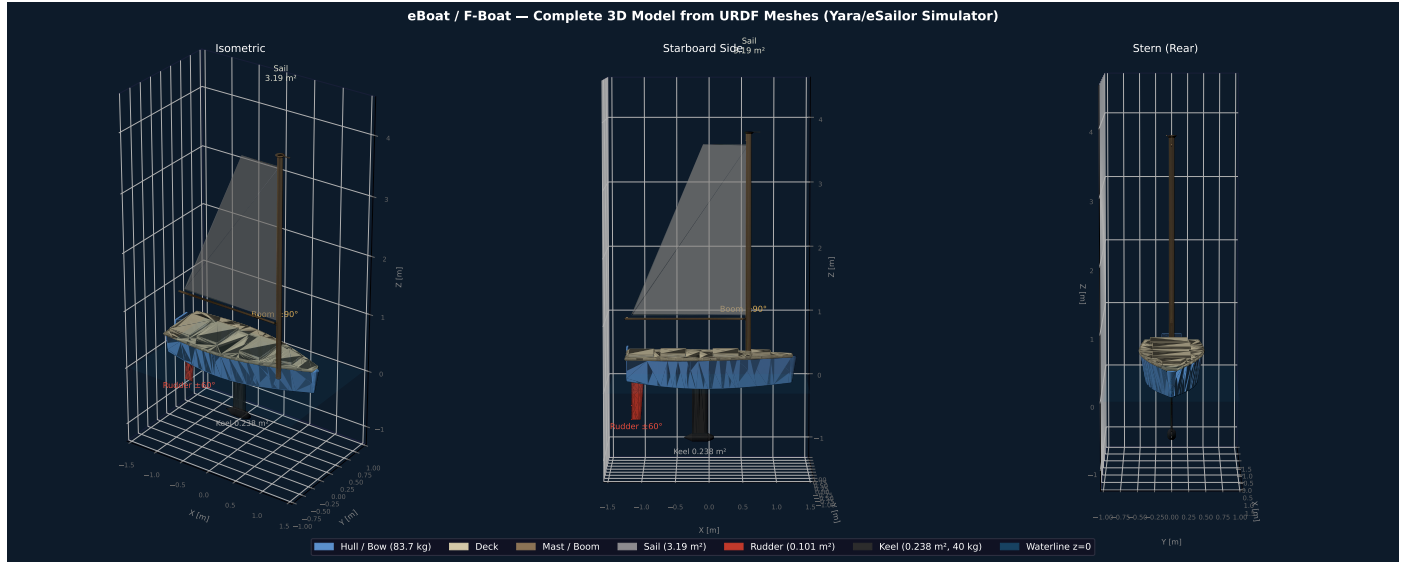


Figure 1: eBoat digital twin (F-Boat) — complete 3D reconstruction from URDF/STL meshes, shown in three views: isometric, starboard, and stern. Key structural elements are color-coded: hull/bow (blue), deck (beige), mast/boom (brown), sail (white, 3.19 m^2), rudder (red, 0.101 m^2), keel (dark gray, 0.238 m^2). Actuated joints (boom $\pm 90^\circ$, rudder $\pm 60^\circ$) are labeled in their respective panels.

Table 2 summarizes the key physical properties extracted directly from the URDF. The vessel’s kinematics are governed by three continuous, normalized actuators:

- **Rudder Control (δ_r):** Normalized to $[-1, 1]$, mapping to a maximum deflection of $\pm 60^\circ$

(± 1.047 rad) for heading management. Joint effort limit: 50 N·m; maximum angular velocity: 0.42 rad/s.

- **Boom Control (δ_b):** Normalized to $[-1, 1]$, controlling the sail aperture up to $\pm 90^\circ$ (± 1.571 rad) to optimize aerodynamic lift. Joint effort limit: 1,000 N·m; maximum angular velocity: 5 rad/s.
- **Auxiliary Propulsion (P_m):** Normalized to $[0, 1]$, representing the power output of the electric motor used in motor-assisted scenarios.

Both the rudder and boom are modeled as Gazebo *revolute joints* actuated directly in angle and torque, i.e., as rotary servo actuators bounded by the joint effort and angular-velocity limits above — not as linear (piston-type) actuators. The commanded action δ_b is applied directly as the boom joint’s target angle; the mainsheet cable and winch that would translate a released cable length into that angle on the physical F-Boat are not separately modeled, since the joint-angle abstraction is kinematically equivalent for the purposes of this thesis. The same applies to the rudder: δ_r is the commanded joint angle, with no intermediate linear-to-rotary transmission stage.

Table 2: Key physical parameters of the eBoat URDF model, as configured in the Gazebo plugins.

Parameter	Value	Component
Hull bounding box	$2.7 \times 1.0 \times 0.63$ m	Collision geometry
Hull mass	83.69 kg	Rigid body
Mast height	3.55 m (cylinder, $r = 0.05$ m)	Mast link
Sail area (A_s)	3.188 m ²	LiftDrag plugin
Rudder area (A_r)	0.101 m ²	LiftDrag plugin
Keel area (A_k)	0.238 m ²	LiftDrag plugin
Air density (ρ_a)	1.201 kg/m ³	Aerodynamic model
Water density (ρ_w)	997.0 kg/m ³	Hydrodynamic model
Sail CoP	(−0.012, 0, 1.5) m	From boom joint
Boom joint limits	$\pm 90^\circ$	Revolute (Z-axis)
Rudder joint limits	$\pm 60^\circ$	Revolute (Z-axis)

2.2.2 Hydrodynamic Model

The eBoat hydrodynamic behavior is governed by a maneuvering model (FOSSEN, 2011) implemented via the `libusv_gazebo_dynamics_plugin` provided by the Yara simulation framework. Using Fossen’s

notation, the equations of motion in the horizontal plane are:

$$\underbrace{(M_{RB} + M_A)}_M \dot{\boldsymbol{\nu}} + C(\boldsymbol{\nu}) \boldsymbol{\nu} + D(\boldsymbol{\nu}) \boldsymbol{\nu} = \boldsymbol{\tau}, \quad (2.1)$$

where $\boldsymbol{\nu} = [u, v, r]^\top$ is the velocity state (surge, sway, yaw rate), M_{RB} is the rigid-body inertia matrix, M_A is the frequency-independent added mass matrix, C is the Coriolis/centripetal matrix, D is the linear and quadratic damping matrix, and $\boldsymbol{\tau} = [\tau_u, \tau_v, \tau_r]^\top$ is the generalized external force vector (sail thrust + rudder force + motor thrust).

The damping matrix $D(\boldsymbol{\nu})$ is parameterized as:

$$D(\boldsymbol{\nu}) = - \begin{bmatrix} X_u + X_{uu}|u| & 0 & 0 \\ 0 & Y_v + Y_{vv}|v| & 0 \\ 0 & 0 & N_r \end{bmatrix}, \quad (2.2)$$

with the values identified empirically for the eBoat hull (Table 3). Wave interaction is modeled by the `ocean_waves` plugin using a Gerstner wave field matched to the visual ocean environment.

Table 3: Identified hydrodynamic drag coefficients as specified in the URDF/Gazebo plugin.

Coefficient	Notation	Value	Type
X_u	Linear surge drag	5.13 N·s/m	Linear
X_{uu}	Quadratic surge drag	7.24 N·s ² /m ²	Quadratic
Y_v	Linear sway drag	24.0 N·s/m	Linear
Y_{vv}	Quadratic sway drag	480.0 N·s ² /m ²	Quadratic
N_r	Yaw damping	120.0 N·m·s/rad	Linear
$\dot{Y}_{\dot{v}}$	Added sway mass	124 kg	Added mass
$\dot{Z}_{\dot{w}}$	Added heave mass	148 kg	Added mass

2.2.3 Sail Aerodynamics and Rudder Hydrodynamics

Propulsive and steering forces are computed by the custom `libLiftDragForces` Gazebo plugin, which models the sail, rudder, and keel as lifting surfaces. For each surface, the lift and drag forces are:

$$F_L = \frac{1}{2} \rho C_L(\alpha) A V_{app}^2, \quad F_D = \frac{1}{2} \rho C_D(\alpha) A V_{app}^2, \quad (2.3)$$

where ρ is the fluid density (air for sail/keel, water for rudder), A is the surface area, V_{app} is the apparent fluid velocity, and α is the attack angle. The lift coefficient $C_L(\alpha)$ and drag coefficient $C_D(\alpha)$ follow thin-airfoil theory, fitted empirically to the F-Boat sail geometry.

The sail's attack angle α_s depends on the current boom joint angle δ_b and the apparent wind angle ϕ_w :

$$\alpha_s = \delta_b - \phi_w \pmod{2\pi}, \quad (2.4)$$

making the sail force strongly sensitive to the boom command. The rudder attack angle is $\alpha_r = \delta_r$ relative to the keel line, with water as the working fluid. The keel passively generates lateral lift that counteracts sway drift.

Figure 2 summarizes the complete aerodynamic and hydrodynamic force model, and Figure 3 shows the resulting polar diagram for the eBoat.

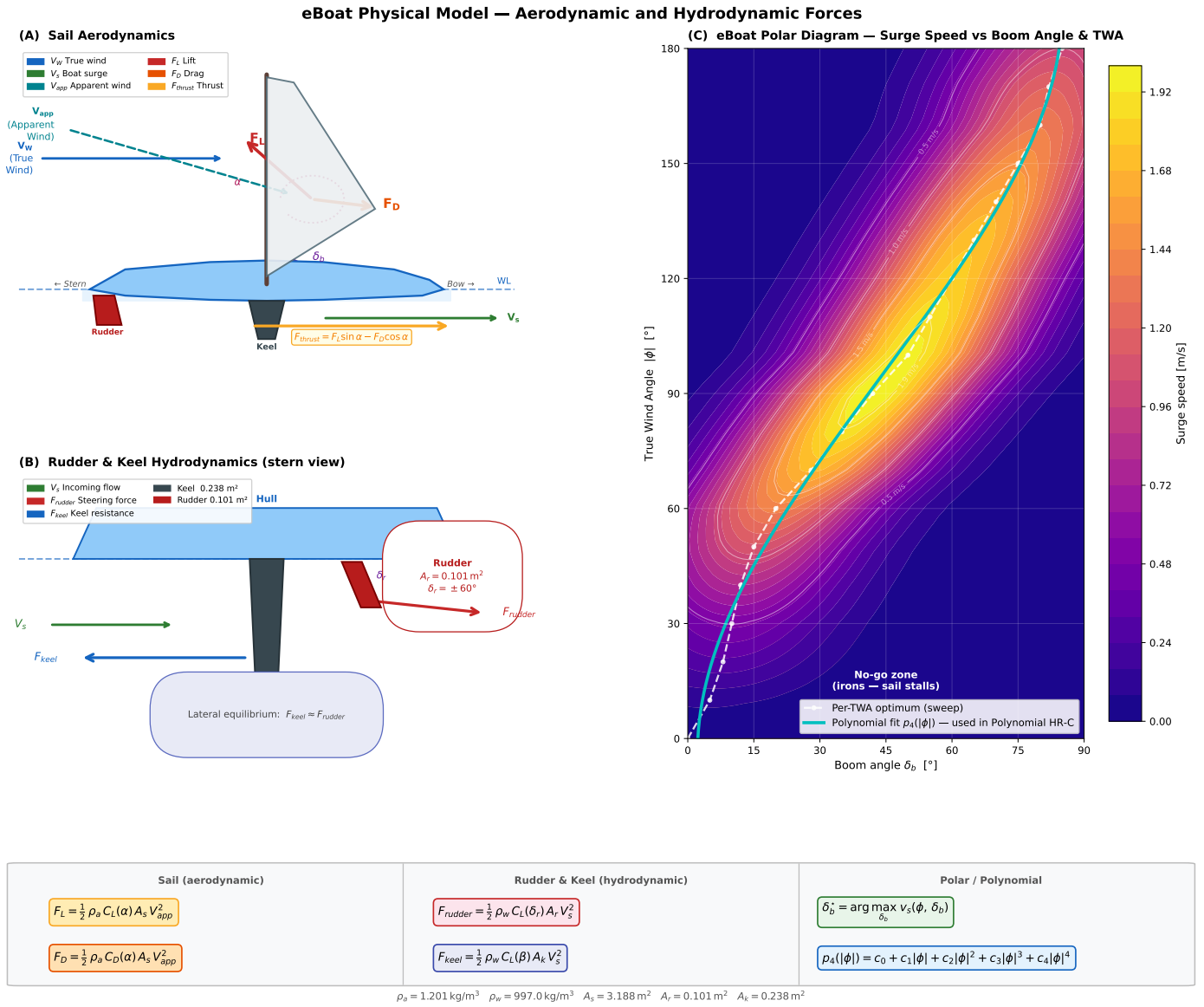


Figure 2: eBoat physics model. **(A)** Sail aerodynamics: true wind V_W , vessel surge V_s , and apparent wind V_{app} combine to produce lift F_L and drag F_D as functions of the boom angle δ_b ; the forward thrust F_{thrust} is the net longitudinal component. **(B)** Rudder and keel hydrodynamics: the rudder (red, $A_r = 0.101 \text{ m}^2$) generates a lateral steering force; the keel (dark gray, $A_k = 0.238 \text{ m}^2$) provides the counteracting sway resistance. **(C)** eBoat polar diagram: contour map of surge speed as a function of boom angle and True Wind Angle (TWA); red dots mark the per-TWA optimum; the white dashed curve is the degree-4 polynomial regressor used by the A*/APF+PPO+Polynomial controller.

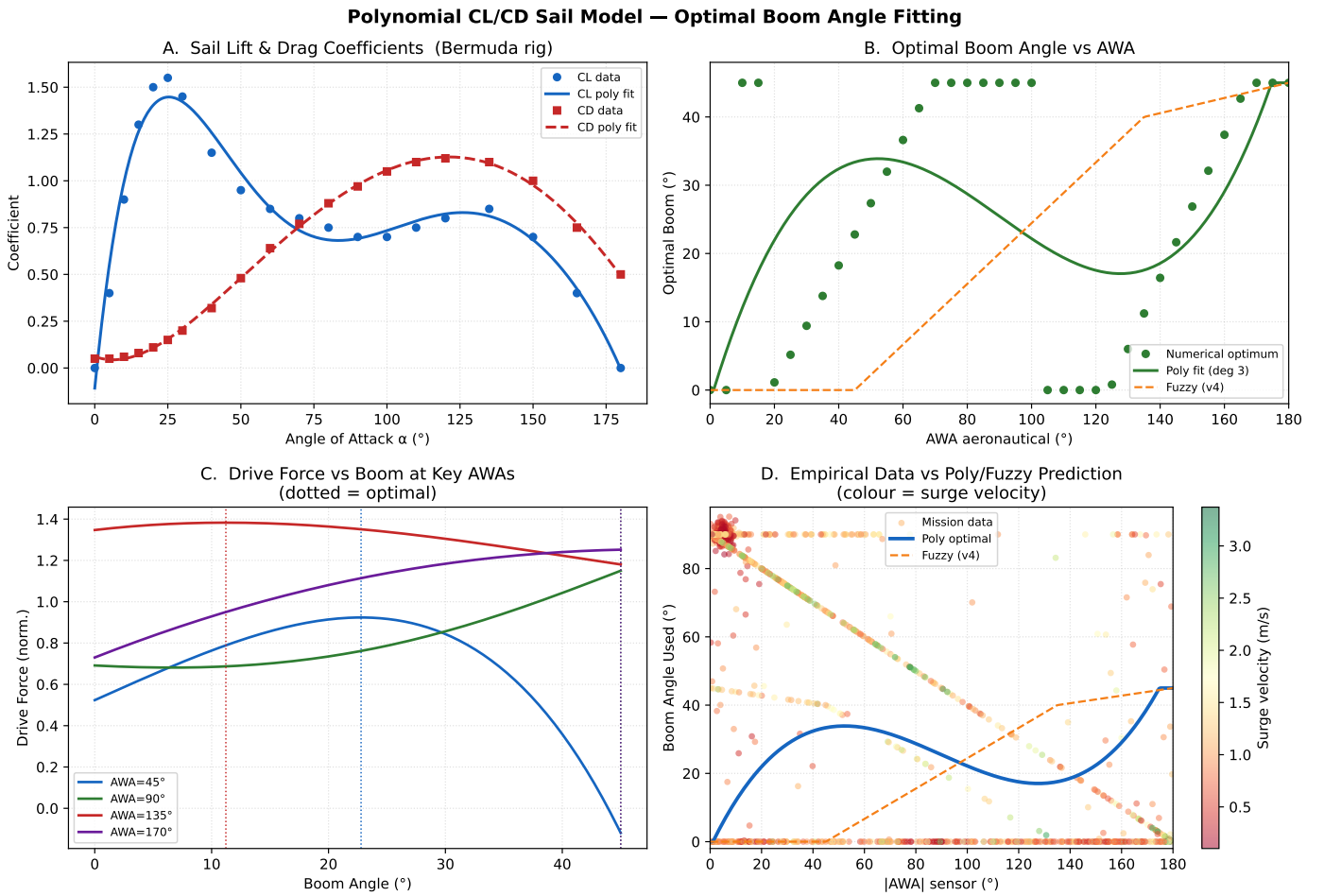


Figure 3: eBoat polar diagram generated from open-loop Gazebo sweeps. Each curve plots surge velocity vs. boom angle at a fixed True Wind Angle. Green dots mark the surge-maximizing boom angle at each TWA; the green curve is the polynomial regressor fitted to these optima (see Section 2.2.3 regarding the fitted degree).

The critical insight from this diagram is that the optimal boom angle is a smooth, continuous function of the wind angle — with no discontinuity to represent, unlike the no-go zone’s stall boundary — even though it is not monotonic over the full $[0^{\circ}, 180^{\circ}]$ range (Figure 3B shows a local maximum near 50° and a local minimum near 110° – 125° before rising again toward a broad reach). This continuity, rather than monotonicity, is what justifies the A*/APF+PPO+Polynomial approach: a single low-degree polynomial curve, without any neural network involvement, can still capture the full shape without the discrete jumps that discretized or rule-based approaches introduce.

2.2.4 Onboard Sensor Suite

The eBoat is equipped with three onboard sensors, all implemented as Gazebo physics-accurate plugins:

1. **2D Planar LiDAR (Hokuyo-style ray sensor):** 720 horizontal beams spanning $\pm 90^\circ$ at 10 Hz, range 0.1–40.0 m, Gaussian noise ($\sigma = 0.01$ m). This sensor is the exclusive source of obstacle proximity information for the HR-C policy; from its full scan, the single scalar $r_{10} = \min(\text{scan})/D_{\text{scan}} - 1$ is extracted and fed into the observation space.
2. **Bow Camera:** 800×800 px, 10 Hz, FOV $\approx 80^\circ$, Gaussian pixel noise ($\sigma = 0.007$). Used for visual monitoring but not included in the RL observation space.
3. **Wind Vane (Biruta):** A physical weathercock modeled via a separate lift-drag plugin ($A = 3.57 \times 10^{-3} \text{ m}^2$), providing the apparent wind angle ϕ_w used in the boom control modules and as observation feature r_5 .

Table 4 summarizes the sensor parameters.

Table 4: eBoat onboard sensor specifications as configured in the URDF/Gazebo model.

Sensor	Type	Rate (Hz)	Range / Resolution	Used in RL?
LiDAR (Hokuyo)	Ray scan	10	0.1–40 m, 720 beams, $\pm 90^\circ$	Yes (r_{10})
Bow Camera	RGB camera	10	800×800 px, FOV 80°	No
Wind Vane	Lift-drag	20	Apparent wind angle ϕ_w	Yes (r_5)

The simulation environment also integrates a dynamic wind plugin (randomized speed 3–9 kn, 10 discrete attack angles) and an ocean current vector field (randomized direction and magnitude per episode), contributing to observation features r_{11} (current-X) and r_{12} (current-Y) and to the net external force τ in Eq. (2.1).

2.3 Other Simulation Environments for Maritime Robotics

Virtual environments are essential for the development of autonomous maritime vehicles, providing a safe and controlled space for testing navigation and control algorithms. Many existing simulators cannot integrate directly with reinforcement learning APIs such as Gymnasium ([FARAMA FOUNDATION](#),

2023) and Stable-Baselines3 (RAFFIN et al., 2021), which limits their utility for developing adaptive control strategies.

The Yara simulator (VASCONCELLOS; NEGREIROS, et al., 2025) addresses these limitations with a comprehensive, validated physics simulation for sailing robots, including a digital twin of the F-Boat with foil-accurate lift and drag dynamics. Yara integrates a ROS Noetic interface, GPS, camera, LiDAR, and wind sensor plugins, and achieves a $300\times$ simulation speedup — enabling 10^6 training steps in approximately eight hours on consumer hardware.

Beyond Yara, a range of simulators has been employed in the maritime robotics community. UWSim and UUV Simulator focus on underwater dynamics and are not designed for surface navigation. USVSim is specialized for surface vehicles and provides a reasonable baseline for control development. MATLAB provides robust numerical computation and is widely used for algorithm development and system modeling (THE MATHWORKS, INC., 2024). The Virtual RobotX (VRX) environment (BINGHAM et al., 2019) emphasizes maritime robotic competitions and multi-sensor integration. Tinoco et al. (2021) (TINOCO; MALHEIRO; SILVA, 2021) presented a Gazebo and ROS-based simulation for an autonomous land yacht equipped with a free-rotating wing sail, demonstrating that polar diagram characterization and aerodynamic plugin integration are feasible outside proprietary tools.

Table 5 summarizes representative simulators found in the literature.

Table 5: Summary of Simulators Used in Related Literature

Article	Simulator	Study Focus
(PARAVISI et al., 2019)	Custom	USV Control in Adverse Conditions
(JIANG et al., 2022)	OpenAI Gym	Collision Avoidance for Autonomous Ships
(RODRIGUEZ, 2021)	American Magic (AM)	Sail Maneuver Optimization
(CHACÓN MOSQUERA, 2021)	Virtual Environment	Autonomous Sailboat Prototype
(MÉXAS; LETA; CLUA, 2022)	Unity	Reinforcement Learning in Sailboats
(BINGHAM et al., 2019)	Virtual RobotX (VRX)	USV Development
(VASCONCELLOS; NEGREIROS, et al., 2025)	Yara (eSailor)	Mapless autonomous sailboat navigation

3 Related Works

This chapter surveys the literature on autonomous maritime navigation, situating the HR-C framework within the broader research landscape. The review proceeds from the most general contributions to the most closely related works. First, broad categories of methods are discussed, citing representative papers within each category. Then, the works that are strictly related to the core problem of this thesis — mapless, sensor-driven navigation for autonomous sailboats — are examined in detail.

3.1 Classical and Metaheuristic Path Planning

Classical path-planning methods for autonomous surface vehicles (ASVs) generally fall into two families: graph-search algorithms and metaheuristic optimizers. Graph-search approaches such as A* and Dijkstra guarantee shortest-path optimality on a discretized environment but require a complete, accurate map of the obstacle field and cannot respond to changes discovered at runtime ([HART; NILSSON; RAPHAEL, 1968](#); [SANTOS et al., 2016](#)). Metaheuristic approaches, including Ant Colony Optimization (ACO), Genetic Algorithms (GA), and Particle Swarm Optimization (PSO), relax the optimality guarantee in exchange for greater flexibility in non-convex search spaces ([DORIGO; MANIEZZO; COLORNI, 1996](#); [ZHANG, J. et al., 2022](#)). Both families share the same structural constraint: planning is performed offline, before execution begins, and any obstacle not represented in the planning grid is invisible to the controller. Several works have explored reactive extensions to these baselines — for instance, adding a proximity-based PID deflection layer to ACO ([ZHENG et al., 2022](#)) — but such extensions handle only slow-changing environments and fail when new obstacles appear suddenly on the planned route.

3.2 Reinforcement Learning and Deep Learning for Maritime Navigation

Reinforcement learning (RL) methods address the map-dependency problem by enabling agents to react to sensory observations at each control step (SUTTON; BARTO, 2018). Deep Reinforcement Learning (DRL) has been applied to obstacle avoidance for ASVs using a variety of formulations: collision avoidance for autonomous ships via risk evaluation on the OpenAI Gym platform (JIANG et al., 2022), adaptive navigation with DRL for dynamic environments (MÉXAS; LETA; CLUA, 2022), real-time obstacle prediction using deep learning (LIU et al., 2022), and COLREGs-compliant dynamic avoidance combining rule-based and optimization approaches (ZHENG et al., 2022). A coarse-to-fine RL strategy for sailboats demonstrated improved adaptability across a wide range of environmental interactions (SMITH et al., 2022).

A complementary line of research has focused on reducing actuator wear and control smoothness under reactive RL policies. High-frequency rudder reversals generated by DRL policies attempting to minimize cross-track error under disturbances cause significant mechanical fatigue (KRAUTWIG et al., 2026). Reward regularization — incorporating penalties for control effort variations — has been proposed as a principled approach to enforce smooth trajectory profiles and preserve hardware integrity over extended deployments (AGYEI; SARHADI; POLANI, 2025).

Hierarchical approaches have proven particularly effective for decomposing long-horizon maritime navigation. Frameworks using rapidly-exploring random trees (RRT) to guide DQN agents accelerate training convergence and improve safety against irregular obstacles (ZHANG, H. et al., 2025). Frameworks that incrementally build sparse local graphs from LiDAR point clouds handle unseen dynamic obstacles without a prior map (SELLERS et al., 2026). A 2025 comprehensive review of AI-based autonomous sailboat navigation specifically identified the need for hybrid, adaptive, and learning-based systems to bridge the gap between idealized simulation and robust real-world deployment (MANKINA et al., 2025).

3.3 Obstacle Avoidance Techniques

Obstacle avoidance methods for ASVs span a wide spectrum of sensing modalities and algorithmic paradigms. Detection-based methods use deep learning to identify obstacles from camera or radar data: Luo (2021) developed a sailboat detection model based on YOLOv5 with attention mechanisms (LUO; NGUYEN; YAN, 2021), while Yang and Shen (2022) proposed millimeter-wave radar for enhanced

detection accuracy in adverse weather (YANG; SHEN, 2022). Planning-based methods include genetic algorithm approaches for real-time collision avoidance and path optimization (ZHANG, J. et al., 2022), and improved velocity obstacle methods for trajectory adjustment under dynamic conditions (WANG et al., 2022). Physics-informed simulation frameworks have also been used to study the robustness of avoidance strategies against realistic wave and current dynamics (ZHOU et al., 2021).

For sailing-specific platforms, Rodriguez et al. (2021) applied adaptive control and machine learning to optimize foiling sailboat maneuvers in a high-fidelity simulator developed for the America’s Cup, demonstrating the applicability of advanced simulation tools to competitive sailing performance (RODRIGUEZ, 2021). Zhang et al. (2022) introduced a sensor-fusion and adaptive control strategy for autonomous sailboats to improve navigation accuracy under environmental perturbations (ZHANG, H. et al., 2022). Mosquera (2021) developed an autonomous sailboat prototype with advanced sensors and a virtual training environment, incorporating computer vision for future collision avoidance (CHACÓN MOSQUERA, 2021).

Table 6 summarizes representative obstacle avoidance techniques from the literature by technique category.

Table 6: Representative Obstacle Avoidance Techniques Categorized by Approach

Category	Representative Works	Key Characteristics
Detection via DL	(LUO; NGUYEN; YAN, 2021), (LIU et al., 2022)	YOLOv5/attention, real-time prediction; sensor-dependent
Radar-based	(YANG; SHEN, 2022)	Millimeter-wave; robust in low visibility
Rule + Optimization	(ZHENG et al., 2022)	COLREGs compliance; limited to predictable environments
Genetic Algorithm	(ZHANG, J. et al., 2022)	Multi-obstacle; real-time but computationally heavy
Velocity Obstacle	(WANG et al., 2022)	Trajectory replanning; assumes known velocities
DRL (End-to-End)	(MÉXAS; LETA; CLUA, 2022), (SMITH et al., 2022)	Adaptive; reward-sensitive; actuator wear concern
HRL + Hierarchical	(ZHANG, H. et al., 2025), (SELLERS et al., 2026)	Decomposed; handles long-horizon and unseen obstacles

3.4 Autonomous Sailboats and Energy-Efficient Maritime Systems

Autonomous sailboat research occupies a specialized niche within the broader ASV literature due to the coupling between propulsion (sail aerodynamics) and maneuverability (rudder hydrodynamics). Cruz et al. (2010) (CRUZ; ALVES, 2010) demonstrated that automatic steering systems for autonomous sailboats can dynamically respond to wind direction and intensity changes, maintaining efficient routes under environmental uncertainty. Energy management is a recurrent concern: Correa et al. (2021) (CORREA et al., 2021) explored restricted Boltzmann machines to optimize energy consumption for long maritime missions. The agent-based simulation work of Smirnov et al. (2021) (SMIRNOV; TOMFORDE, 2022) modeled realistic maritime environments to study autonomous ferry behavior, highlighting the complexity of multi-agent maritime traffic modeling.

This thesis also builds on the candidate’s own earlier platform work, which is related to this research area but does not itself address the mapless path-planning problem tackled here. S. Júnior et al. (2013) (S. JÚNIOR et al., 2013) introduced N-BOAT, an early autonomous robotic sailboat platform, and Araújo et al. (2013) (ARAÚJO; AROCA; GONÇALVES, 2013) modeled the steering and control of an early autonomous boat; Santos et al. (2016) (SANTOS et al., 2016) subsequently detailed the design and implementation of a control system for a sailboat robot, a direct predecessor of the F-Boat platform used throughout this thesis. More recently, Negreiros et al. (2022) (NEGREIROS et al., 2022) presented sustainable sea-monitoring solutions built around the N-Boat and F-Boat platform twins, and Araújo et al. (2024) (ARAÚJO, André Paulo Dantas et al., 2024) detailed the general system architecture and COTS prototyping of an AIoT-enabled sailboat for autonomous aquatic ecosystem monitoring, reinforcing the application-level motivation described in Chapter 1.

Adversarial and falsification-based evaluation has gained traction as a methodology for stress-testing autonomous maritime systems. Müller et al. (2025) (MÜLLER et al., 2025) applied falsification-driven RL to generate adversarial training scenarios that expose rule compliance failures in DRL agents, establishing a principled test methodology for maritime autonomy. This approach is aligned with the empirical falsification tests conducted in this thesis.

3.5 Strictly Related Works

The works most directly related to this thesis combine mapless navigation, hierarchical control, and autonomous sailing in a simulation-based evaluation framework.

Vasconcellos et al. (2025) (VASCONCELLOS; NEGREIROS, et al., 2025) presented the Yara

simulation environment, the direct platform for all experiments in this thesis. Yara provides a physics-validated digital twin of the F-Boat robotic sailboat with foil-accurate dynamics and a $300\times$ speedup for RL training, addressing the gap between simplified grid worlds and real maritime conditions. The present thesis extends this infrastructure with a hierarchical controller, an APF-augmented observation space, explicit sail actuation modules, and a five-scenario evaluation campaign.

Ruan et al. (2023) (RUAN; DAI; YU, 2023) and Wang et al. (2023) (WANG; JIANG; WANG, 2023) proposed hierarchical reinforcement control frameworks for autonomous navigation in complex environments, validating the architectural decomposition into strategic and tactical layers that HR-C adopts. While those works target ground robots and general ASVs, respectively, this thesis adapts and extends the hierarchical paradigm to the aerodynamic constraints of robotic sailing.

Mankina et al. (2025) (MANKINA et al., 2025) conducted a comprehensive review of AI-based autonomous sailboat navigation, identifying the lack of mapless, sensor-driven methods as a critical research gap. The HR-C framework directly addresses this gap with an empirical demonstration across progressively adversarial scenarios. Müller et al. (2025) (MÜLLER et al., 2025) contributed a falsification-driven evaluation methodology that this thesis adopts in Scenario C (unknown-obstacle falsification test), providing a principled adversarial baseline.

Zhang et al. (2025) (ZHANG, H. et al., 2025) demonstrated that using a global planner (RRT) to guide a local DRL agent (DQN) significantly improves training convergence and safety, a result that strongly supports the A*-guided PPO design of HR-C. Sellers et al. (2026) (SELLERS et al., 2026) showed that constructing sparse local graphs from LiDAR point clouds enables mapless navigation in environments with unseen dynamic obstacles — corroborating the sensor-driven, mapless premise of this thesis and suggesting future directions for HR-C’s high-level component.

The key distinction of this thesis relative to all prior works is the simultaneous validation of mapless navigation, actuator mechanical stress quantification, dynamic multi-vessel generalization, and explicit sail actuation module design — all within a single unified framework, evaluated against classical baselines under controlled falsification conditions.

4 Architectural Design and Implementation of the HR-C Framework

This chapter presents the mathematical model and architectural formalization of the Hierarchical Reinforcement Control (HR-C) framework, together with the implementation of the baseline algorithms used for comparison. The core innovation of HR-C resides in the decoupling of strategic goal-seeking from tactical obstacle avoidance, achieved through a 12-dimensional observation space and a multi-level reinforcement learning structure. The evaluation scenarios and experimental setup are described in Chapter 5.

4.1 Hierarchical Layer Taxonomy

To keep every algorithm name in this thesis unambiguous, this section defines the three independent layers of the navigation stack up front and states explicitly which combinations were implemented and evaluated. Every algorithm name follows the convention *[high-level layer]/[low-level layer](+[sail layer])*, extending the classical baseline notation already standard in the literature (e.g., A*+PID).

High-level (strategic) layer. Produces a coarse route or waypoint sequence, before or independently of any low-level reactive correction.

- **A***: offline graph search on a static occupancy grid (Section 4.2.1); obstacle-blind at runtime.
- **A*(RP)**: A* re-invoked at runtime whenever the vessel deviates more than 5 m from its planned path, or a LiDAR return falls within 4 m (Section 4.2.1).
- **ACO**: offline metaheuristic search on the same occupancy grid (Section 4.2.2).
- **A*/APF**: A* on a permanently *empty* grid, producing only sparse directional waypoints; the Artificial Potential Field (Section 4.5) fuses those waypoints with live LiDAR repulsion into a

single normalized guidance vector consumed downstream. This is the high-level layer used by every HR-C configuration evaluated in this thesis.

Low-level (tactical) layer. Converts the high-level guidance into a rudder command at every control step.

- **PID:** cross-track error controller (Section 4.2.1); ignores LiDAR and never receives the 12-dimensional observation space.
- **PPO:** the reinforcement-learned policy (Section 4.4), driven by the full 12-dimensional observation space, including the A*/APF guidance vector.

Sail-actuation layer (optional third layer). Determines the boom command. This layer only exists downstream of the PPO low-level controller: the PID baselines in this thesis hold the boom fixed and rely on the auxiliary motor for propulsion (Algorithm 4.1), so they were never combined with a sail module.

- **Implicit:** no separate module — the PPO network outputs the boom command directly, alongside rudder and motor (Section 4.2.3).
- **Fuzzy:** a Mamdani fuzzy inference system computes the boom command from wind angle and speed (Section 4.9.1); PPO retains only rudder and motor.
- **Polynomial:** a closed-form polar-diagram regressor computes the boom command (Section 4.9.2); PPO retains only rudder and motor.

Layers are *not* combined arbitrarily: Table 7 lists every combination actually implemented, its resulting algorithm name, and whether it consumes the 12-dimensional observation space. Every algorithm evaluated in Chapter 5 corresponds to exactly one row.

4.2 Comparative Algorithm Implementation

Three distinct core navigation paradigms were implemented for the comparative study. Each baseline was configured to achieve its best possible performance within its architectural constraints, ensuring that observed differences in results reflect genuine algorithmic limitations rather than under-tuning.

Table 7: Every high-level $\times$ low-level $\times$ sail-layer combination implemented in this thesis. A dash indicates the layer is not applicable (the PID baselines hold the boom fixed and never receive the 12-D observation vector).

Algorithm name	High-level	Low-level	Sail layer	Uses 12-D obs.?
A*+PID	A*	PID	— (fixed boom)	No
A*(RP)+PID	A*(RP)	PID	— (fixed boom)	No
ACO+PID	ACO	PID	— (fixed boom)	No
A*/APF+PPO	A*/APF	PPO	Implicit	Yes
A*/APF+PPO+Fuzzy	A*/APF	PPO	Fuzzy	Yes
A*/APF+PPO+Polynomial	A*/APF	PPO	Polynomial	Yes

The theoretical foundations of A*, ACO, PID, and PPO are reviewed in Chapter 2; this section details the specific parameterization and integration decisions made for this thesis.

4.2.1 A*+PID (Non-Reactive Baseline)

The A*+PID baseline represents the classical deterministic approach. A* computes the globally optimal path offline, prior to mission start, using a discretized occupancy grid. The PID controller then tracks this pre-computed path, issuing rudder commands proportional to the cross-track error. The PID ignores all LiDAR data during execution: if an unmapped obstacle appears on the route, the controller has no structural mechanism to react.

Grid parameters. The simulation environment occupies a 200×200 m bounded arena. A* operates on a 200×200 cell occupancy grid with 1 m cell resolution. Each cell within a 2 m radius of a registered obstacle centroid is marked as occupied. The Euclidean distance heuristic $h(n) = \|n - g\|_2$ is used, guaranteeing optimality (HART; NILSSON; RAPHAEL, 1968).

PID heading controller. The cross-track error e_t is defined as the signed perpendicular distance from the vessel’s position to the line segment connecting the current waypoint pair. The rudder command is:

$$\delta_r = K_p e_t + K_i \sum_{\tau=0}^t e_\tau \Delta t + K_d \frac{e_t - e_{t-1}}{\Delta t}, \quad (4.1)$$

with gains $K_p = 1.2$, $K_i = 0.05$, $K_d = 0.08$, and a control period $\Delta t = 0.05$ s. Rudder commands are saturated at $\pm 30^\circ$ as a software limit (the hardware joint allows $\pm 60^\circ$, but a conservative bound reduces overshoot in low-speed sailing). A waypoint is considered reached when the Euclidean distance

falls below $\varepsilon_{wp} = 3$ m.

Algorithm 4.1 formalizes the heading PID execution loop used by both A*+PID and ACO+PID at runtime.

Algorithm 4.1 PID Heading Controller Runtime Loop

Require: Planned path $\mathcal{W}$, gains K_p, K_i, K_d , waypoint tolerance ε_{wp}

Ensure: Rudder command δ_r at each control step

```

1:  $k \leftarrow 0$ ;  $e_{\text{int}} \leftarrow 0$ ;  $e_{\text{prev}} \leftarrow 0$ 
2: while  $k < |\mathcal{W}|$  do
3:    $p \leftarrow \text{GETPOSITION}()$ 
4:   if  $\|p - \mathcal{W}_k\| \leq \varepsilon_{wp}$  then
5:      $k \leftarrow k + 1$ ; continue
6:   end if
7:    $e_t \leftarrow \text{CROSSTRACKERROR}(p, \mathcal{W}_{k-1}, \mathcal{W}_k)$  ▷ signed perp. distance
8:    $e_{\text{int}} \leftarrow e_{\text{int}} + e_t \cdot \Delta t$ 
9:    $\dot{e} \leftarrow (e_t - e_{\text{prev}}) / \Delta t$ 
10:   $\delta_r \leftarrow K_p e_t + K_i e_{\text{int}} + K_d \dot{e}$ 
11:   $\delta_r \leftarrow \text{clip}(\delta_r, -30^\circ, +30^\circ)$  ▷ software saturation
12:   $\text{APPLYACTUATORS}(\delta_r, 90^\circ, P_m)$  ▷ boom fixed open; motor from mission mode
13:   $e_{\text{prev}} \leftarrow e_t$ 
14: end while

```

A*(RP)+PID variant. A replanning variant (A*(RP)+PID) was also implemented, in which A* is re-invoked whenever the vessel deviates more than 5 m from the planned path or whenever a new LiDAR return falls within 4 m. This variant is included in Scenario D and the competitive race to represent the most capable map-based baseline. Even with replanning, the method cannot represent moving agents in its occupancy grid.

4.2.2 ACO+PID (Reactive-Layer Baseline)

The ACO+PID baseline represents metaheuristic planning augmented with a minimal reactive safety layer. The global route is computed by Ant Colony Optimization on the same 200×200 grid used by A*. At runtime, the PID heading controller tracks the ACO-generated path, and a separate proximity trigger deflects the rudder when any LiDAR beam falls below a threshold distance. However, the global ACO path is never updated in response to LiDAR data, so the method shares the same structural blind spot as A*+PID when facing unmapped obstacles.

ACO parameters. The implementation follows the Ant System formulation (DORIGO; MANIEZZO; COLORNI, 1996). A colony of $m = 100$ artificial ants constructs solutions over the discretized grid. Pheromone trails τ_{ij} are initialized uniformly to $\tau_0 = 0.1$ and updated at the end of each iteration. The movement probability from node i to node j is:

$$p_{ij} = \frac{[\tau_{ij}]^\alpha [\eta_{ij}]^\beta}{\sum_{k \in \mathcal{N}_i} [\tau_{ik}]^\alpha [\eta_{ik}]^\beta}, \quad (4.2)$$

where $\eta_{ij} = 1/d_{ij}$ is the inverse-distance heuristic, $\alpha = 1$ weights pheromone, and $\beta = 2$ weights the heuristic. Global pheromone evaporation is applied after each iteration with rate $\rho = 0.1$:

$$\tau_{ij} \leftarrow (1 - \rho) \tau_{ij} + \sum_{k=1}^m \Delta \tau_{ij}^k, \quad (4.3)$$

where $\Delta \tau_{ij}^k = Q/L_k$ if ant k used edge (i,j) (with $Q = 1$, L_k the total path length of ant k), and 0 otherwise. The algorithm runs for 100 iterations before mission start.

PID reactive layer. The PID controller uses the same gains as A*+PID ($K_p = 1.2$, $K_i = 0.05$, $K_d = 0.08$). An additional proximity deflection term is superimposed on the PID output when the minimum LiDAR return $d_{\min}$ falls below 5 m:

$$\delta_r^{\text{ACO}} = \delta_r^{\text{PID}} + K_{\text{rep}} \text{sgn}(\phi_{\text{obs}}) \cdot \max\left(0, 1 - \frac{d_{\min}}{5}\right), \quad (4.4)$$

where ϕ_{obs} is the bearing of the nearest obstacle relative to the vessel heading and $K_{\text{rep}} = 15^\circ$. This deflection is purely reactive and does not update the ACO global plan.

4.2.3 A*/APF+PPO (Proposed Mapless Architecture)

The hierarchical framework operates without any prior map: A* plans over an empty grid to provide directional waypoints, and the PPO low-level controller issues actuator commands based solely on real-time LiDAR observations augmented by the APF guidance vector. This architecture serves as the structural skeleton for the three A*/APF+PPO variants evaluated in this thesis (Implicit, Fuzzy, Polynomial), which differ only in which component controls the boom actuator. Figure 4 illustrates the hierarchical interaction; the full implementation details — observation space, APF formulation, reward function, PPO hyperparameters, and curriculum training — are presented in the following sections of this chapter.

Hierarchical Reinforcement Control (HR-C) Process Flow

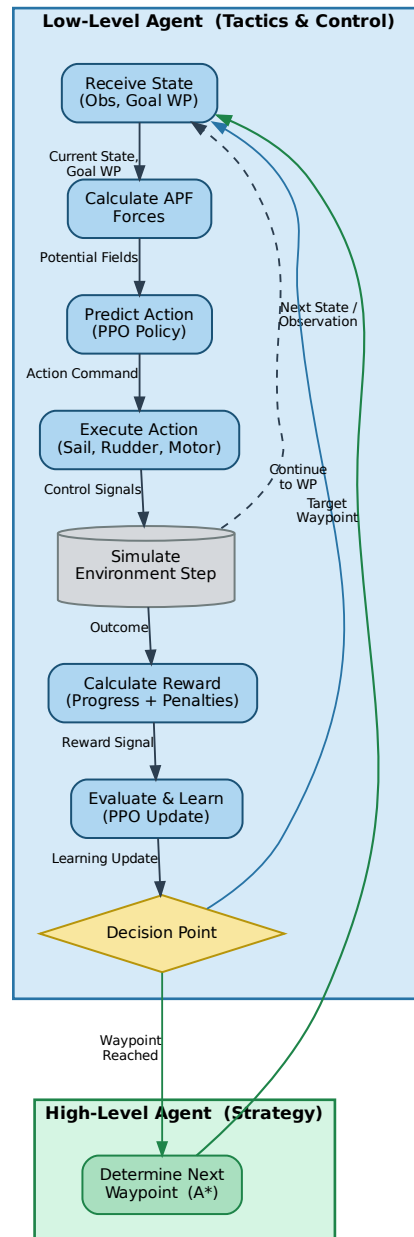


Figure 4: Hierarchical Reinforcement Control (HR-C) process flow diagram, illustrating the interaction between the High-Level (Strategy: A*/APF) and Low-Level (Tactics & Control: PPO) components.

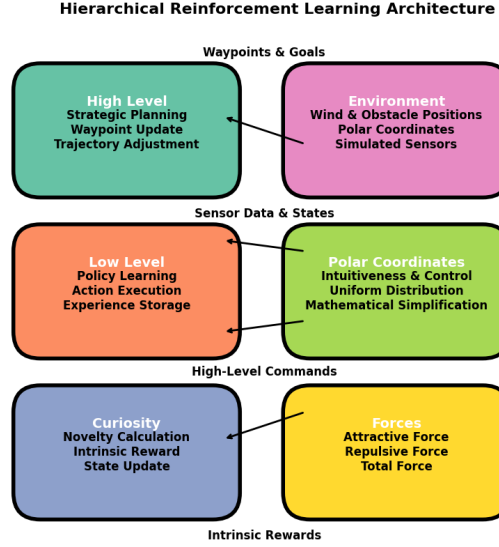


Figure 5: Hierarchical Reinforcement Control Architecture Block Diagram, detailing the data flow from sensors to actuator outputs.

4.3 The Hierarchical Architecture

The HR-C architecture (illustrated in Figure 5) is divided into two distinct logical layers that operate at different timescales:

The High-Level Agent defines the global directional intent. Using a coarse A* search on an empty or highly abstracted grid, it generates a sequence of target waypoints $\mathbf{w}_i$. The agent monitors the Euclidean distance $d_e = \|\mathbf{q} - \mathbf{w}_i\|$ and switches to the next waypoint once $d_e \leq \varepsilon_{wp}$.

The Low-Level Agent is the primary controller, trained via Proximal Policy Optimization (PPO). It receives the current target waypoint from the high-level layer and real-time sensory inputs, outputting continuous commands for the rudder and boom.

4.4 12-Dimensional Observation Space Engineering

To enable mapless navigation, the agent's perception is modeled as a 12-dimensional state vector $\mathcal{O}_t$. Each component is normalized to the $[-1, 1]$ or $[0, 1]$ range to ensure stable gradient updates during training. A key design choice is that the navigational guidance features r_1 and r_2 do **not** encode raw waypoint geometry; instead, they encode the *resultant* APF vector $\hat{\mathbf{F}}_{apf}$ (Section 4.5), which already

combines waypoint attraction with LiDAR-derived obstacle repulsion. This encoding allows the neural network to directly exploit mathematically pre-processed guidance rather than learning to combine raw sensor signals from scratch:

1. r_1 : Magnitude of the normalized APF resultant vector $\|\mathbf{F}_{apf}\|$, clipped to $[0, 1]$. Encodes the net urgency of the combined attractive and repulsive field.
2. r_2 : Bearing of $\hat{\mathbf{F}}_{apf}$ relative to the vessel heading, normalized by $180^\circ \in [-1, 1]$. Replaces the raw geometric bearing to the waypoint with an APF-guided direction that already deflects around obstacles.
3. r_3 : Normalized apparent wind speed (AWS), clipped to $[0, 1]$.
4. r_4 : Boat surge velocity, normalized by maximum expected surge.
5. r_5 : Apparent wind angle (normalized by 180°).
6. r_6 : Current boom angle (normalized by 90°). Removed for explicit sail module variants (11-dim vector).
7. r_7 : Current rudder angle (normalized by 60°).
8. r_8 : Auxiliary motor power usage, normalized to $[0, 1]$.
9. r_9 : Previous bearing to waypoint (normalized by 180°), included to penalize oscillatory heading behavior.
10. r_{10} : **Minimum LiDAR distance**: $r_{10} = \min(\mathcal{S}_t)/D_{\text{scan}} - 1 \in [-1, 0]$. The primary proximity signal; approaches -1 as the vessel closes on an obstacle.
11. r_{11} : Ocean current X-component (normalized by maximum expected magnitude).
12. r_{12} : Ocean current Y-component (normalized by maximum expected magnitude).

Table 8 summarizes the observation space, including the normalization range and physical source of each feature.

Table 8: 12-Dimensional Observation Space of the A*/APF+PPO Agent. Features r_1 – r_2 encode the APF resultant vector; r_{10} encodes LiDAR proximity. Feature r_6 is removed for the Fuzzy and Polynomial sail-layer configurations (11-dim).

Idx	Description	Range	Source	Role
r_1	APF resultant magnitude $\ \mathbf{F}_{apf}\ $	[0,1]	APF (Alg. 4.2)	Navigation urgency
r_2	APF resultant bearing (rel. heading)	[−1,1]	APF (Alg. 4.2)	Obstacle-deflected target direction
r_3	Apparent wind speed (AWS)	[0,1]	Wind vane sensor	Sail trim context
r_4	Surge velocity	[−1,1]	IMU / state estimator	Speed feedback
r_5	Apparent wind angle ϕ_w	[−1,1]	Wind vane sensor	Aerodynamic context
r_6	Boom angle δ_b (<i>Implicit only</i>)	[−1,1]	Actuator state	Sail self-awareness
r_7	Rudder angle δ_r	[−1,1]	Actuator state	Heading self-awareness
r_8	Motor power P_m	[0,1]	Motor driver	Energy context
r_9	Previous bearing to waypoint	[−1,1]	State history	Oscillation damping
r_{10}	Min. LiDAR distance ($\min \mathcal{S}/D_{\text{scan}} - 1$)	[−1,0]	Hokuyo scan	Proximity alarm
r_{11}	Ocean current X -component	[−1,1]	Current plugin	Drift correction
r_{12}	Ocean current Y -component	[−1,1]	Current plugin	Drift correction

4.5 Feature Engineering: APF as Guidance

A critical design choice in this thesis is the utilization of Artificial Potential Fields (APF) as an observation feature rather than a deterministic controller. The APF component calculates a virtual guidance vector $\mathbf{F}_{APF}$:

$$\mathbf{F}_{APF} = \mathbf{F}_{att} + \sum_i \mathbf{F}_{rep,i}, \quad (4.5)$$

where the attractive force $\mathbf{F}_{att} = k_{att}(\mathbf{w} - \mathbf{q})/\|\mathbf{w} - \mathbf{q}\|$ pulls toward the active waypoint $\mathbf{w}$, and each repulsive contribution is:

$$\mathbf{F}_{rep,i} = -k_{rep} \left(\frac{1}{d_i} - \frac{1}{d_{inf}} \right) \frac{1}{d_i^2} \hat{\mathbf{n}}_i, \quad d_i < d_{inf}, \quad (4.6)$$

with d_i being the i -th LiDAR beam range, d_{inf} the influence radius, and $\hat{\mathbf{n}}_i$ the beam direction. This vector is fed into the PPO observation space, allowing the neural network to learn a policy that weighs mathematical guidance against the complex aerodynamic constraints of sailing (such as the no-go zone). Algorithm 4.2 formalizes the computation.

4.6 Integration of Ocean Currents

Ocean currents are implemented as an external force vector acting on the boat’s hydrodynamic model within the Gazebo ocean plugin. The current field is characterized by a speed magnitude v_c and a direction angle θ_c (both randomized per episode during training). These are decomposed into Cartesian

Algorithm 4.2 APF Guidance Vector from Real-Time LiDAR Scan

Require: Boat position $\mathbf{q}$, waypoint $\mathbf{w}$, LiDAR scan $\mathcal{S} = \{(d_i, \theta_i)\}$, influence radius $d_{inf} = 10$ m, gains k_{att}, k_{rep}

Ensure: Normalized guidance vector $\hat{\mathbf{F}}_{apf}$

```

1:  $\mathbf{F}_{att} \leftarrow k_{att} (\mathbf{w} - \mathbf{q}) / \|\mathbf{w} - \mathbf{q}\|$ 
2:  $\mathbf{F}_{rep} \leftarrow \mathbf{0}$ 
3: for each beam  $(d_i, \theta_i) \in \mathcal{S}$  do
4:   if  $d_i < d_{inf}$  then
5:      $\hat{\mathbf{n}}_i \leftarrow (\cos \theta_i, \sin \theta_i)$ 
6:      $m_i \leftarrow k_{rep} (\frac{1}{d_i} - \frac{1}{d_{inf}}) / d_i^2$ 
7:      $\mathbf{F}_{rep} \leftarrow \mathbf{F}_{rep} - m_i \hat{\mathbf{n}}_i$ 
8:   end if
9: end for
10:  $\mathbf{F}_{apf} \leftarrow \mathbf{F}_{att} + \mathbf{F}_{rep}$ 
11: return  $\mathbf{F}_{apf} / \|\mathbf{F}_{apf}\|$ 

```

body-frame force components:

$$F_{c,x} = \rho_w C_{dc} A_{hull} v_c^2 \cos(\theta_c - \psi), \quad F_{c,y} = \rho_w C_{dc} A_{hull} v_c^2 \sin(\theta_c - \psi), \quad (4.7)$$

where $\rho_w = 997 \text{ kg/m}^3$ is the water density, C_{dc} is the hull drag coefficient for current loading, A_{hull} is the effective lateral hull area, and ψ is the vessel's yaw angle. These components enter the generalized force vector $\boldsymbol{\tau}$ in the equations of motion (Eq. 2.1) as additive disturbances, physically displacing the vessel regardless of actuator commands.

To enable the PPO agent to actively compensate for current-induced drift, the current vector components are included in the observation space as features r_{11} and r_{12} (Table 8). This allows the neural network to learn current-aware heading corrections without requiring explicit inverse-dynamics computation.

Figure 6 provides a geometric conceptual illustration of how the attractive APF force, obstacle repulsion, and ocean current combine into the net resultant vector that informs the HR-C agent's heading decision.

4.7 Reward Function Formalization

The policy was optimized using an extrinsic reward function designed to balance efficiency and safety. The reward at each step t is defined as:

$$R_t = \text{clip}(R_{\text{progress}} + R_{\text{safety}} + R_{\text{wind}}, -1, 1), \quad (4.8)$$

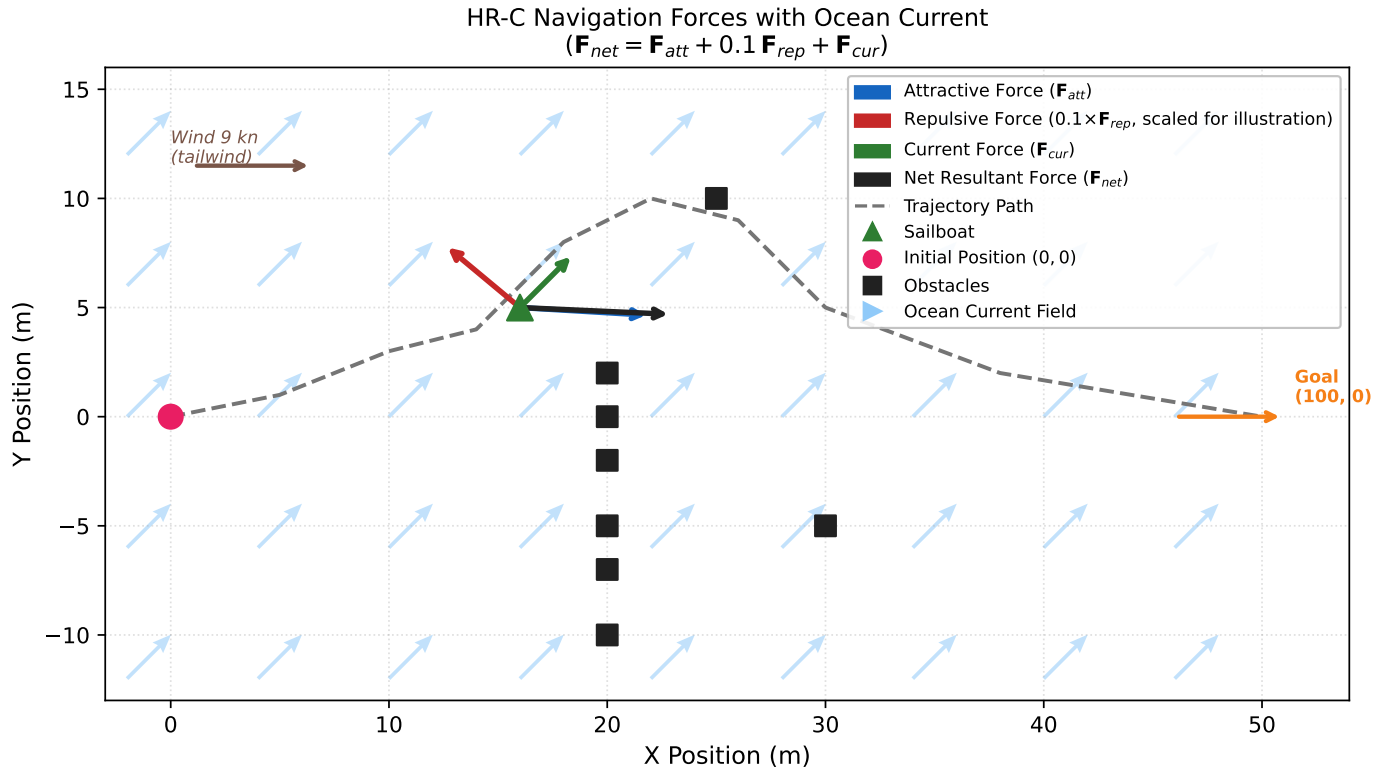


Figure 6: Conceptual illustration of the individual forces acting on the HR-C agent: attractive force (blue), scaled repulsive force (red, $0.1 \times F_{rep}$), ocean current force (green), and the resulting net resultant vector (black).

with the three components:

- **Progress Reward:** $R_{\text{progress}} = 5 \times (d_{t-1} - d_t)/D_{\text{MAX}}$, rewarding direct convergence toward the waypoint. Negative values penalize regression.
- **Safety Penalty:** $R_{\text{safety}} = -1.0$ if $d_{\text{obs}} < 2$ m (imminent collision); -0.5 if $2 \leq d_{\text{obs}} < 5$ m (proximity warning).
- **Wind Bonus:** $R_{\text{wind}} = +0.05$ if the absolute apparent wind angle $|\phi_{\text{wind}}| \in [30^\circ, 150^\circ]$, promoting aerodynamic sailing over motor usage.

Algorithm 4.3 Step Reward Computation

Require: Previous distance d_{t-1} , current distance d_t , closest obstacle d_{obs} , wind angle ϕ_{wind} , waypoint reached flag *done*

Ensure: Scalar reward R_t

```

1:  $R_{\text{progress}} \leftarrow 5.0 \times (d_{t-1} - d_t)/D_{\text{MAX}}$ 
2: if  $d_{\text{obs}} < 2.0$  then
3:    $R_{\text{safety}} \leftarrow -1.0$ 
4: else if  $d_{\text{obs}} < 5.0$  then
5:    $R_{\text{safety}} \leftarrow -0.5$ 
6: else
7:    $R_{\text{safety}} \leftarrow 0.0$ 
8: end if
9: if  $|\phi_{\text{wind}}| \in [30^\circ, 150^\circ]$  then
10:   $R_{\text{wind}} \leftarrow +0.05$ 
11: else
12:   $R_{\text{wind}} \leftarrow 0.0$ 
13: end if
14: if done then
15:  return  $+1.0$    (waypoint-reached terminal bonus)
16: end if
17: return  $\text{clip}(R_{\text{progress}} + R_{\text{safety}} + R_{\text{wind}}, -1.0, 1.0)$ 

```

Algorithm 4.3 details the complete step-by-step computation, including the waypoint-reached terminal bonus. Table 9 shows the PPO hyperparameters used for all A*/APF+PPO variants. They are configured by way of using the `stable-baselines3` library.

4.8 Curriculum Learning and Robustness Training

To address the sim-to-real gap, the agent was trained via a four-phase curriculum that progressively introduced real-world uncertainties. Phase 1 used 3-knot winds (calm) to establish basic waypoint

Table 9: PPO Hyperparameters used for all A*/APF+PPO variants. Configured via the `stable-baselines3` library.

Hyperparameter	Value	Rationale
Total timesteps	5×10^5	Curriculum over 4 phases
Steps per rollout (T)	2048	Long horizon for ocean dynamics
Mini-batch size	64	Stable gradient variance
Epochs per update (K)	10	Maximize data reuse
Learning rate (α)	3×10^{-4}	Adam optimizer default
Discount (γ)	0.99	Long-range credit assignment
GAE λ	0.95	Bias-variance trade-off
Clip range (ε)	0.2	Standard PPO bound
Entropy coefficient	0.01	Prevents premature collapse
Value loss coefficient	0.5	Balanced actor-critic
Max grad norm	0.5	Gradient clipping for stability
Policy network	MLP [64, 64]	Sufficient for 12-D observation
Activation	Tanh	Bounded outputs

convergence. Subsequent phases introduced randomized wind speeds (up to 9 knots), 10 randomized attack angles, and variable ocean currents. This progressive complexity ensured that the agent learned to compensate for drift and aerodynamic stall, producing a policy robust enough for the evaluation scenarios presented in Chapter 5.

Table 10 summarizes the four training phases. Algorithm 4.4 and Algorithm 4.5 formalize the PPO training loop and the complete HR-C runtime control loop, respectively.

Table 10: Four-phase curriculum training schedule. Wind speed and angle are randomized within the shown ranges at each episode reset.

Phase	Wind Speed	Wind Angle	Obstacles	Currents	Timesteps
1	3 kn (fixed)	180° (fixed)	None (empty field)	None	100 000
2	3–5 kn	{0, 90, 180, 270}	4 static cylinders	None	150 000
3	3–9 kn	10 random angles	8 static cylinders	Randomized	150 000
4	3–9 kn	10 random angles	8 static + dynamic agents	Randomized	100 000
Total					500 000

4.9 Explicit Sail Actuation Modules

A key architectural extension investigated in this thesis is the structural *decoupling* of the sail actuator from the PPO network — introducing the sail-actuation layer defined in Section 4.1 as a third, op-

Algorithm 4.4 PPO Training Loop (Proximal Policy Optimization)**Require:** Environment $\mathcal{E}$, policy π_θ , value function V_ϕ , hyperparameters $T, K, \varepsilon, \alpha, \gamma, \lambda$ **Ensure:** Trained parameters (θ^*, ϕ^*)

```

1: for each training iteration do
2:   Collect rollout buffer  $\mathcal{B}$  of  $T$  steps from  $\pi_\theta$  in  $\mathcal{E}$ 
3:   Compute returns  $\hat{R}_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k}$ 
4:   Compute GAE advantages:  $\hat{A}_t = \sum_{k=0}^{\infty} (\gamma\lambda)^k \delta_{t+k}$ , where  $\delta_t = r_t + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$ 
5:   for epoch = 1, ...,  $K$  do
6:     for each mini-batch  $\mathcal{M} \subset \mathcal{B}$  do
7:        $r_t(\theta) \leftarrow \pi_\theta(a_t|s_t) / \pi_{\theta_{\text{old}}}(a_t|s_t)$  (prob. ratio)
8:        $L_{\text{clip}} \leftarrow \mathbb{E} \left[ \min \left( r_t \hat{A}_t, \text{clip}(r_t, 1-\varepsilon, 1+\varepsilon) \hat{A}_t \right) \right]$ 
9:        $L_{\text{vf}} \leftarrow \mathbb{E} \left[ (V_\phi(s_t) - \hat{R}_t)^2 \right]$ 
10:       $L_{\text{ent}} \leftarrow \mathbb{E} [\mathcal{H}[\pi_\theta(\cdot|s_t)]]$ 
11:       $L \leftarrow -L_{\text{clip}} + c_1 L_{\text{vf}} - c_2 L_{\text{ent}}$ 
12:       $\theta \leftarrow \theta - \alpha \nabla_\theta L$ ;  $\phi \leftarrow \phi - \alpha \nabla_\phi L_{\text{vf}}$ 
13:    end for
14:  end for
15:   $\theta_{\text{old}} \leftarrow \theta$ 
16: end for
17: return  $\theta^*, \phi^*$ 

```

Algorithm 4.5 HR-C Runtime Inference Loop**Require:** Start p_0 , goal g , trained policy π_θ , sail module SAILMODULE (or NULL)**Ensure:** Actuator commands (δ_r, δ_b, m) at every step t

```

1:  $\mathcal{W} \leftarrow \text{ASTARPLAN}(p_0, g, M_\emptyset)$  ▷ empty map — no obstacle knowledge
2:  $k \leftarrow 0$ ;  $p \leftarrow p_0$ 
3: while  $\|p - g\| > \varepsilon_{\text{goal}}$  do
4:   if  $\|p - \mathcal{W}_k\| \leq \varepsilon_{wp}$  then  $k \leftarrow \min(k + 1, |\mathcal{W}| - 1)$ 
5:   end if
6:    $\mathbf{F}_{apf} \leftarrow \text{APF}(p, \mathcal{W}_k, \mathcal{S}_t)$  ▷ Alg. 4.2
7:   Assemble observation  $\mathbf{o}_t = [r_1, \dots, r_{12}]$  (see Section 4.4)
8:    $[\tilde{\delta}_r, \tilde{\delta}_b, \tilde{m}] \leftarrow \pi_\theta(\mathbf{o}_t)$  ▷ PPO forward pass
9:   if SAILMODULE  $\neq$  NULL then
10:     $\delta_b \leftarrow \text{SAILMODULE}(\phi_{\text{wind}}, \text{AWS})$  ▷ Fuzzy or Polynomial
11:   else
12:     $\delta_b \leftarrow \tilde{\delta}_b$  ▷ A*/APF+PPO (Implicit): network controls boom
13:   end if
14:   Apply  $(\tilde{\delta}_r, \delta_b, \tilde{m})$  to simulator
15:    $p \leftarrow \text{GETPOSITION}()$ ;  $t \leftarrow t + 1$ 
16: end while

```

tional layer of the hierarchy. In the baseline $A^*/APF+PPO$ (Implicit sail layer, described above), the boom command δ_b is one of the three PPO outputs, forcing the network to simultaneously optimize aerodynamic propulsion and reactive obstacle avoidance. This conflation degrades sample efficiency and produces high-frequency boom oscillations that propagate as rudder jitter through hydrodynamic coupling.

To solve this, two explicit sail modules are introduced as drop-in replacements for the boom output. Both modules expose a unified interface: given the True Wind Angle (ϕ) and Apparent Wind Speed (AWS), they return a deterministic reference boom angle $\delta_b^* \in [0^\circ, 90^\circ]$. The PPO policy is correspondingly reduced to a 2-dimensional action vector $[\delta_r, m]$, delegating all aerodynamic control to the module.

4.9.1 Fuzzy Sail Controller ($A^*/APF+PPO+Fuzzy$)

The Fuzzy module employs a Mamdani-type Fuzzy Inference System (FIS) encoding classical sailing heuristics. Input 1 maps the absolute wind angle $|\phi|$ onto five overlapping triangular membership functions (**irons**, **close-hauled**, **beam**, **broad**, **run**); Input 2 maps AWS into **light**, **medium**, and **strong** intensities. A 15-rule base (Table 11) translates these inputs into the output boom trim linguistic states (**tight**, **mid-tight**, **mid-loose**, **loose**), defuzzified via the center-of-gravity method. The controller is interpretable, requires no training data, and functions immediately at runtime.

Table 11: Representative subset of the 15-rule Mamdani fuzzy rule base. Each cell shows the output boom trim. Blank cells are “don’t-care” combinations.

Wind Angle \ Speed	Light	Medium	Strong
irons ($0^\circ\text{--}45^\circ$)	tight	tight	tight
close-hauled ($30^\circ\text{--}75^\circ$)	tight	tight	mid-tight
beam ($60^\circ\text{--}110^\circ$)	mid-tight	mid-loose	mid-loose
broad ($100^\circ\text{--}150^\circ$)	mid-loose	mid-loose	loose
run ($140^\circ\text{--}180^\circ$)	loose	loose	loose

Algorithm 4.6 formalizes the Mamdani inference and center-of-gravity defuzzification.

4.9.2 Polynomial Sail Controller ($A^*/APF+PPO+Polynomial$)

The Polynomial module fits a closed-form regressor directly to the vessel’s empirical polar diagram:

$$\delta_b^* = p_k(|\phi|) = \sum_{i=0}^k c_i |\phi|^i, \quad k = 4, \quad (4.9)$$

Algorithm 4.6 Fuzzy Sail Controller Inference (Mamdani FIS)**Require:** True Wind Angle ϕ , Apparent Wind Speed AWS**Ensure:** Boom angle $\delta_b^* \in [0^\circ, 90^\circ]$

```

1: // Fuzzification
2:  $\mu_{\text{angle}}[j] \leftarrow \text{trimf}(|\phi|, \text{params}[j])$  for  $j \in \{\text{irons}, \text{close}, \text{beam}, \text{broad}, \text{run}\}$ 
3:  $\mu_{\text{speed}}[k] \leftarrow \text{trimf}(\text{AWS}, \text{params}[k])$  for  $k \in \{\text{light}, \text{medium}, \text{strong}\}$ 
4: // Rule evaluation (Mamdani min-AND)
5: activations  $\leftarrow []$ 
6: for each rule  $\mathcal{R}_i$ : IF angle= $j$  AND speed= $k$  THEN trim= $c_i$  do
7:    $\mu_i \leftarrow \min(\mu_{\text{angle}}[j], \mu_{\text{speed}}[k])$ 
8:   activations.append( $\mu_i, c_i$ )
9: end for
10: // Defuzzification (center of gravity)
11:  $\delta_b^* \leftarrow \frac{\sum_i \mu_i \cdot c_i}{\sum_i \mu_i}$ 
12: return clip( $\delta_b^*, 0^\circ, 90^\circ$ )

```

where the coefficients $\mathbf{c} = [c_0, \dots, c_4]$ are obtained offline by sweeping $\phi \in [0, \pi]$ and $\delta_b \in [0^\circ, 90^\circ]$ in open-loop simulations and recording the boom angle that maximizes surge speed at each wind angle. The degree $k = 4$ is chosen by cross-validated MSE. During inference the output is clipped to actuator limits $[0^\circ, 90^\circ]$.

Algorithm 4.7 details the offline fitting procedure and Algorithm 4.8 the runtime inference.

4.9.3 Integration and Architectural Impact

With the boom delegated to the explicit module, the PPO observation space shrinks from 12 to 11 dimensions (the current boom angle feature r_6 is removed), and the network can focus exclusively on spatial heading regulation and reactive LiDAR-based obstacle avoidance. The practical consequence—confirmed by the competitive race experiments in Chapter 5—is a simultaneous improvement in navigation speed and a dramatic reduction in rudder wear, breaking the assumed speed–energy Pareto trade-off.

4.10 Computational Complexity Analysis

This section quantifies the time and memory cost of every component in Table 7, separating *training-time* cost (paid once, offline) from *runtime* cost (paid at every control step during deployment) — the two regimes the complexity of an onboard navigation system must be assessed in.

Algorithm 4.7 Polynomial Sail Controller — Offline Polar-Diagram Fitting

Require: Simulator $\mathcal{E}$, angle grid $\Phi = \{0, \Delta\phi, \dots, 180^\circ\}$, boom grid $\mathcal{B} = \{0, \Delta b, \dots, 90^\circ\}$, candidate degrees $\mathcal{K} = \{2, 3, 4, 5, 6\}$, folds $F = 5$

Ensure: Polynomial coefficients $\mathbf{c}^*$, optimal degree k^*

```

1: for each  $\phi \in \Phi$  do
2:   for each  $\delta_b \in \mathcal{B}$  do
3:     Set wind angle  $\phi$ , boom  $\delta_b$ ; run  $\mathcal{E}$  for  $T_{\text{obs}}$  steps
4:      $v(\phi, \delta_b) \leftarrow$  mean surge velocity
5:   end for
6:    $\delta_b^{\text{opt}}[\phi] \leftarrow \arg \max_{\delta_b} v(\phi, \delta_b)$ 
7: end for
8:  $X \leftarrow \Phi$ ;  $y \leftarrow [\delta_b^{\text{opt}}[\phi] \mid \forall \phi \in \Phi]$ 
9: for each degree  $k \in \mathcal{K}$  do
10:   $\text{CV-MSE}[k] \leftarrow F\text{-fold cross-validated MSE of } \text{polyfit}(X, y, k)$ 
11: end for
12:  $k^* \leftarrow \arg \min_k \text{CV-MSE}[k]$ 
13:  $\mathbf{c}^* \leftarrow \text{polyfit}(X, y, k^*)$ 
14: return  $\mathbf{c}^*$ 

```

Algorithm 4.8 Polynomial Sail Controller — Runtime Inference

Require: True Wind Angle ϕ , coefficients $\mathbf{c} = [c_0, \dots, c_{k^*}]$

Ensure: Boom angle $\delta_b^* \in [0^\circ, 90^\circ]$

```

1:  $x \leftarrow |\phi|$ 
2:  $\delta_b^* \leftarrow \sum_{i=0}^{k^*} c_i \cdot x^i$ 
3: return  $\text{clip}(\delta_b^*, 0^\circ, 90^\circ)$ 

```

4.10.1 Classical Baselines: A* and ACO

A* searches an occupancy grid with V cells and, on an 8-connected grid, up to $E \approx 8V$ directed edges. With a binary-heap priority queue, a single search runs in $O(E \log V)$ time and $O(V)$ space (open/closed-set membership, and the g -score, f -score, and parent-pointer arrays needed to reconstruct the path). For the 200×200 m arena at 1 m resolution used throughout Chapter 5, $V = 40\,000$ cells, giving a worst-case bound of $E \log_2 V \approx 320\,000 \times 15.3 \approx 4.9 \times 10^6$ node expansions and roughly 1 MB of working memory per search — negligible at this scale, and the reason A*+PID and A*(RP)+PID run comfortably in real time in these experiments.

The concern raised regarding A*'s memory cost becomes material at *operational* scale, not at the scale of these controlled experiments. Occupancy-grid memory grows quadratically with the linear size of the covered area at fixed resolution: a real long-endurance mission covering, e.g., a 5×5 km bay at the same 1 m resolution requires $V = 25 \times 10^6$ cells, roughly $625\times$ this thesis's test arena, pushing working memory for the score arrays alone into the hundreds of megabytes and each individual search into tens of millions of node expansions. A*(RP)+PID additionally re-runs this full search from scratch at every replanning trigger (Section 4.1), multiplying that cost by the number of triggers encountered during a mission. This is precisely the structural scalability limit that motivates HR-C's high-level layer running A* on a permanently *empty* grid (Table 7): with no obstacle cells to store, its memory footprint is $O(1)$ in the number of obstacles and grows only with the sparse waypoint sequence, independent of how finely a real deployment area would otherwise need to be discretized.

ACO explores $n_{\text{ants}} \times n_{\text{iter}}$ probabilistic path constructions per planning call ($100 \times 100 = 10\,000$ in this thesis's configuration), each traversing on the order of the path length in grid steps, plus an $O(E)$ pheromone matrix τ_{ij} maintained in memory (≈ 2.6 MB for the same 200×200 m grid). This makes a full ACO planning call substantially more expensive in wall-clock time than a single A* search, for a solution that is not guaranteed to be optimal (Section 3.1).

4.10.2 A*/APF+PPO: Training and Runtime Cost

Runtime (deployment) cost. The PPO policy network is a small multilayer perceptron, MLP [64, 64], mapping a 12-dimensional (or 11-dimensional, Section 4.9) observation to actuator commands (Table 9). Its forward pass costs on the order of $2 \times (12 \times 64 + 64 \times 64 + 64 \times 3) \approx 10^4$ floating-point operations — sub-microsecond on any modern CPU, and, critically, *independent of the size of the operational area or the number of obstacles present*, since the policy consumes only the fixed-size APF-fused observation vector rather than a raw occupancy grid. This is the central computational argument for the mapless

design: runtime cost for A*/APF+PPO is $O(1)$ per control step regardless of mission scale, whereas A*+PID's and ACO+PID's replanning cost (when replanning is enabled) grows with the size of the occupancy grid as discussed above.

Training cost. Training runs for 5×10^5 timesteps across the four curriculum phases (Table 10), with rollouts of $T = 2048$ steps, $K = 10$ epochs per update, and minibatches of 64 (Table 9), yielding on the order of $(5 \times 10^5 / 2048) \times 10 \times (2048 / 64) \approx 7.8 \times 10^4$ gradient updates on a network of a few thousand parameters — again computationally trivial relative to the cost of stepping the physics simulation itself, which dominates wall-clock training time. Chapter 2 reports that Yara sustains roughly 10^6 PPO timesteps in about eight hours on consumer hardware; at that same rate, this thesis's 5×10^5 -timestep curriculum is expected to train in approximately four hours, a figure driven chiefly by Yara's $300\times$ simulation speedup rather than by the network's (already negligible) computational cost. Unlike the grid-based baselines, this training cost does not need to be paid again as the deployment area grows — the trained policy is reused unchanged regardless of the size or shape of the operational region, only the sparse A*/APF waypoint sequence generated at runtime differs.

5 Experiments and Results

This chapter presents the comparative experiments conducted in the Yara/eSailor simulation environment (VASCONCELLOS; NEGREIROS, et al., 2025). The evaluation is structured across five scenarios of progressively increasing realism, stripping away idealized assumptions to expose the operational capabilities of the tested algorithms. Standard fully-mapped conditions (Scenario A) are followed by actuator mechanical stress under sail-only mode (Scenario B), an empirical falsification test with unmapped obstacles (Scenario C), generalization to dynamic vessel obstacles (Scenario D), and a competitive multi-agent race benchmarking explicit sail actuation strategies (Scenario E). A statistical robustness analysis using $N = 9$ independent runs per algorithm concludes the chapter.

5.1 Experimental Setup and Evaluation Scenarios

A relevant contribution of this thesis is the establishment of five complementary evaluation scenarios designed to systematically test the hypotheses formulated in Chapter 1. Scenarios A through D cover increasingly adversarial conditions for the planner and Scenario E introduces the competitive multi-agent race benchmark that evaluates explicit sail actuation strategies. Not every algorithm is evaluated in every scenario: Table 12 states exactly which ones are, and why, to avoid the false impression that the tested set changes arbitrarily from section to section.

Table 12: Algorithms evaluated in each scenario. A*(RP)+PID is withheld from Scenario C by design (Section 5.4); the Fuzzy and Polynomial sail modules only exist downstream of the PPO low-level controller (Table 7) and are therefore evaluated only in Scenario E, where they are the object of the comparison.

Scenario	A*+PID	A*(RP)+PID	ACO+PID	A*/APF+PPO	+Fuzzy	+Polynomial
A — Standard map	Yes	—	Yes	Yes	—	—
B — Sail-only stress	Yes	—	Yes	Yes	—	—
C — Falsification	Yes	<i>excluded by design</i>	Yes	Yes	—	—
D — Dynamic patrol	—	Yes	Yes	Yes	—	—
E — Competitive race	Yes	Yes	Yes	Yes	Yes	Yes

5.1.1 Scenario A: Standard Navigation (Baseline Comparison)

The agent must navigate from (0,0) to a waypoint at (100,0) m under a 9-knot easterly tailwind, with 8 static obstacles clustered between 20 and 30 meters. This scenario establishes the baseline efficiency metrics (step count, path length, and mean speed) when classical planners have access to a perfect map.

5.1.2 Scenario B: Sail-Only Actuator Stress Analysis

Scenario B reuses Scenario A’s map, obstacles, and wind exactly, changing only one variable: motor propulsion is disabled ($P_m = 0$). It is kept as a distinct scenario, rather than folded into A or C, precisely *because* it isolates that single variable — with the motor available, all methods can (and do) mask poor sail control by leaning on auxiliary thrust, as seen in A*+PID’s low 6.2° mean boom angle in Scenario A (Table 13). Removing the motor is what makes the raw aerodynamic actuation effort, and therefore the wear it produces, directly measurable. This scenario isolates the relationship between reactive navigation and physical hardware wear. Three metrics evaluate the actuators:

- **Total Angular Displacement:** $\sum |\Delta\theta|$, the sum of all actuator movements.
- **Direction Reversals (N_{rev}):** The count of mechanical load cycles.
- **RMS Impulse Amplitude:** The root-mean-square of incremental displacements, indicating the aggressiveness of the control policy.

5.1.3 Scenario C: Unknown-Obstacle Robustness (Falsification Test)

The ultimate test for mapless autonomy involves placing a physical obstacle at (50,0) m directly on the path, without updating the planning grids of the A* or ACO baselines. This scenario is designed to empirically falsify the plan-then-execute paradigm by demonstrating its inability to respond to unmapped environmental anomalies.

5.1.4 Scenario D: Dynamic Multi-Vessel Environment (Moving Obstacles)

The fourth scenario tests A*/APF+PPO in the most realistic setting considered in this thesis: a navigation corridor where six SDF-physical vessels (rigid bodies: $4.0 \times 1.8 \times 1.4$ m), following scripted sinusoidal and straight-line trajectories, patrol autonomously at 20 Hz, creating intermittent, unpredictable blockages. These vessels are detected by the real LiDAR sensor — no artificial data injection is performed — making each avoidance event a genuine sensor-to-action loop. The wind is set to 4.63 kn from 180° (Beaufort 2). Eight static cylindrical obstacles remain in the corridor. Neither A* nor ACO can represent the positions of moving obstacles in their planning grids, rendering map-based replanning structurally useless.

5.1.5 Scenario E: Explicit Sail Actuation Benchmark (Competitive Race)

To rigorously evaluate the explicit sail controllers proposed in Chapter 4 (Fuzzy and Polynomial modules), a competitive multi-agent benchmark is established. Six sailboats — each driven by a different control stack, including the classical baselines and three A*/APF+PPO variants — are spawned simultaneously from adjacent positions along the Y axis. They race to a common sequence of waypoints under identical 5 kn/ 180° wind conditions, in an environment containing both the 8 static obstacles and the 6 dynamic patrolling vessels from Scenario D. This scenario standardizes the evaluation of speed, safety, and mechanical stress under extreme multi-obstacle conditions.

5.2 Performance Under Standard Conditions (Idealized Map)

In this baseline scenario, classical planners (A*+PID and ACO+PID) were granted perfect a priori knowledge of the environment (a 9-knot easterly tailwind with 8 static obstacles). Auxiliary motor propulsion was permitted. Table 13 and Figure 7 summarize the primary quantitative metrics.

Table 13: Quantitative Performance Summary (Standard Scenario with Perfect Map Knowledge). This table reports navigation efficiency; mean boom angle is included only as a qualitative indicator of sail vs. motor reliance, *not* as a mechanical-wear metric. The wear-relevant metrics — total angular displacement $\sum |\Delta\theta|$, reversal count, and RMS — are the sum of absolute movements recommended for stress comparison, and are reported for this same idealized-map condition under sail-only propulsion in Table 14 (Scenario B) and across all algorithms in Figure 32 (Section 5.7).

Method	Steps	Path (m)	Mean v (m/s)	Mean Boom (°)	Clearance (m)	Map Required?
A*+PID	26	109.8	2.17	6.2	0.29	Yes
ACO+PID	100	266.1	1.34	19.2	0.27	Yes
A*/APF+PPO	74	202.7	1.39	17.9	0.27	No

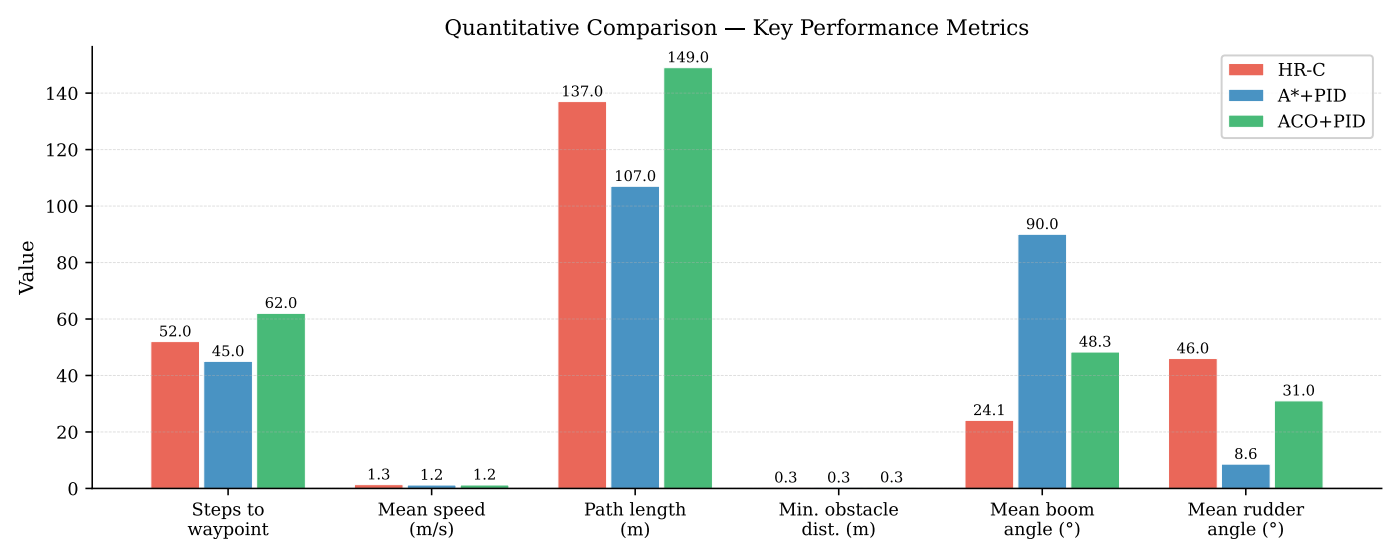


Figure 7: Grouped bar chart of key performance metrics for the standard scenario. A*+PID dominates on speed and efficiency only because it uniquely relies on a fully known environment.

While A*+PID appears to dominate in path efficiency and speed, this performance is entirely contingent upon a premise rarely met in open-water navigation: a static, perfectly mapped world. Furthermore, its mean boom angle of 6.2° indicates an over-reliance on auxiliary motor propulsion, ignoring aerodynamic lift. A*/APF+PPO, conversely, completed the mission utilizing **no prior information**, calculating all avoidance maneuvers reactively via LiDAR and actively utilizing the sail (17.9° boom angle).

5.2.1 Trajectory and Convergence Analysis

Figure 8 illustrates the spatial paths. The A*+PID trajectory reflects the benefit of offline optimization. A*/APF+PPO exhibits a smooth, reactive trajectory that dynamically bypasses the obstacle cluster. ACO+PID generates a highly suboptimal arc (-51 m deviation), compounding the inefficiencies of stochastic search.

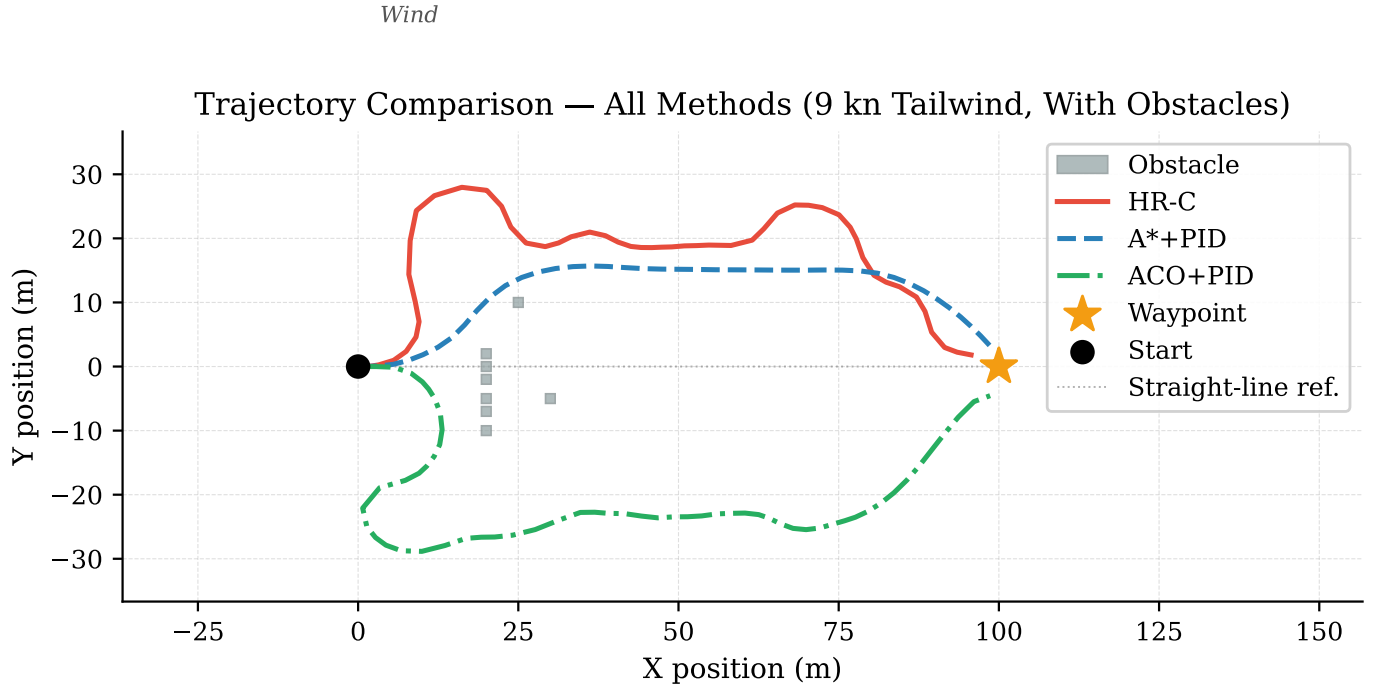


Figure 8: XY trajectories of the three navigation methods under 9-knot tailwind conditions with static obstacles. The dashed grey line represents the theoretical optimal straight line.

The convergence profile (Figure 9) highlights the behavioral differences. While A*+PID converges monotonically, A*/APF+PPO's non-monotonic profile reflects its reactive nature: distance temporarily increases as the agent executes APF-driven avoidance maneuvers before reconverging.

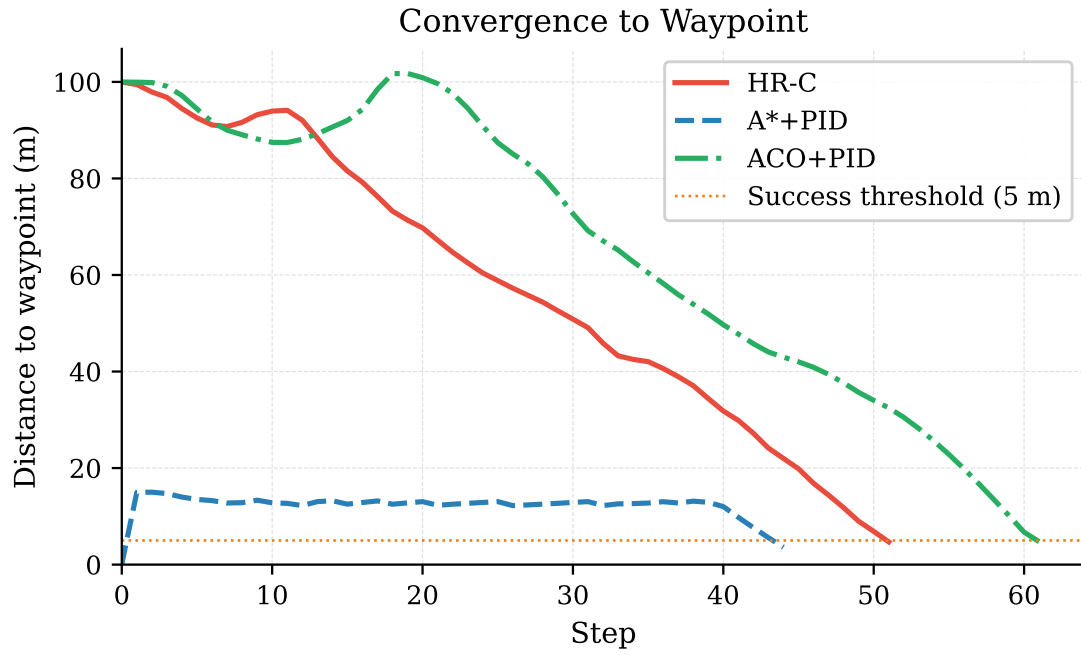


Figure 9: Distance to the target waypoint as a function of elapsed simulation steps. A*/APF+PPO’s non-monotonic profile is the mathematical signature of real-time reactive obstacle avoidance.

5.2.2 Obstacle Clearance and Control Effort

Figure 10 shows that all methods briefly enter the 2-meter danger zone near the obstacle cluster without colliding. However, A*/APF+PPO achieves this safety *autonomously* through learned PPO control, whereas classical methods rely on pre-computed safe routes. Figure 11 presents violin plots of boom and rudder angle distributions, confirming that A*/APF+PPO’s greater variance reflects active, sensor-driven corrections rather than passive tracking.

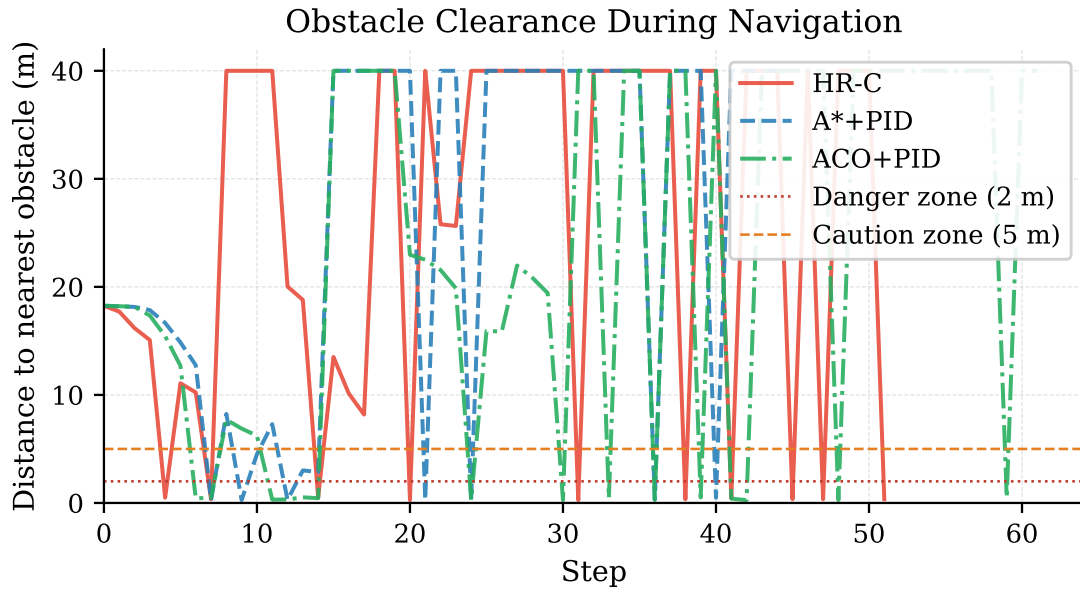


Figure 10: Distance to the nearest obstacle at each simulation step. A*/APF+PPO recovers from proximity events dynamically through APF repulsion.

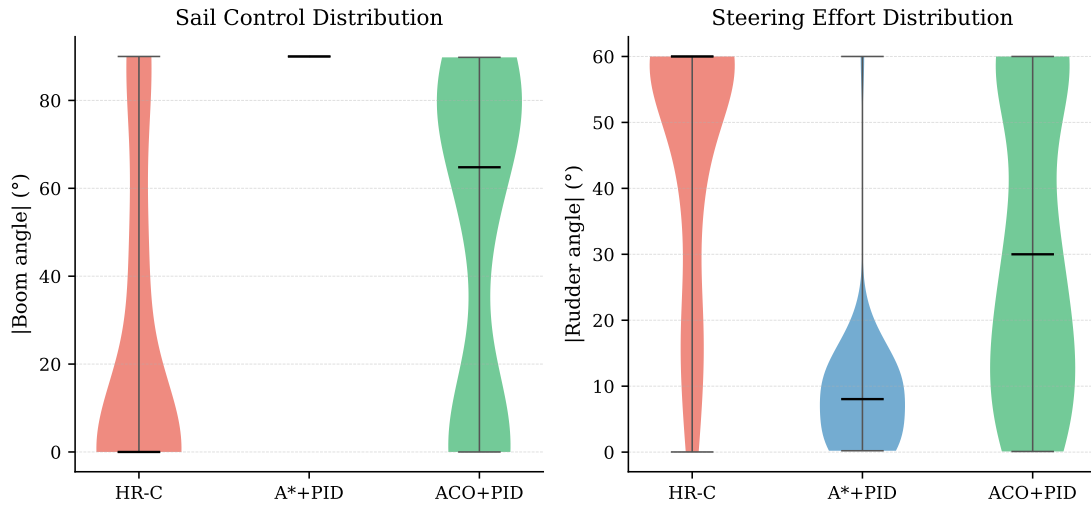


Figure 11: Violin plots of absolute boom and rudder angle distributions. The greater variance in A*/APF+PPO demonstrates active, sensor-driven corrections in response to LiDAR readings.

5.2.3 Wind Utilization and Overall Performance (Scenario A)

Figure 12 confirms that A*/APF+PPO uses varying apparent wind angles (mean -11.7°) to generate aerodynamic lift during lateral maneuvers, unlike the rigid head-on approach of classical PID

controllers.

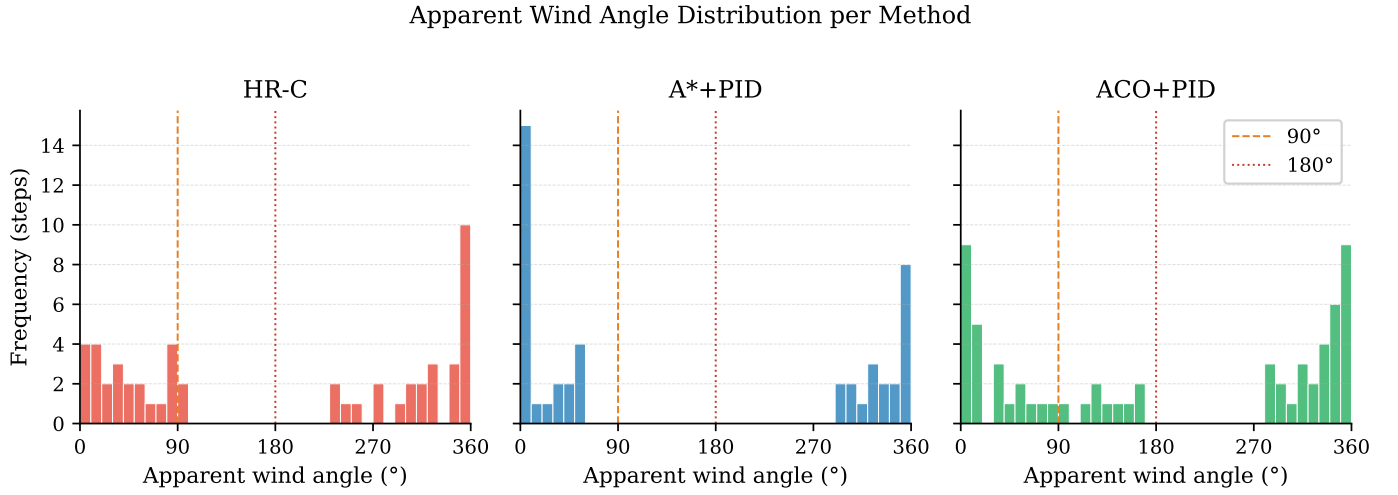


Figure 12: Apparent wind angle histograms demonstrating A*/APF+PPO’s active aerodynamic sailing.

To close the analysis of Scenario A, the radar chart (Figure 13) normalizes performance across standard metrics. While A*+PID dominates idealized conditions, this evaluation omits the unknown-obstacle robustness, which conclusively disqualifies map-dependent methods for real-world deployment; that robustness gap is quantified in Scenario C, after the sail-only stress analysis of Scenario B (Section 5.4). A*/APF+PPO provides the only balanced, deployable solution for genuinely autonomous marine operations.

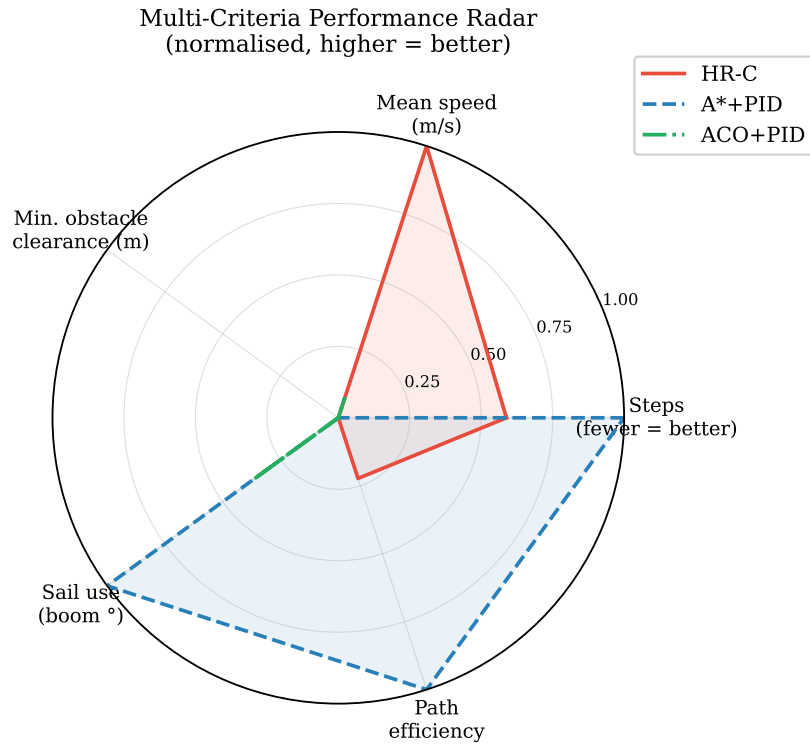


Figure 13: Radar chart comparing methods across normalized criteria under known-environment conditions.

5.3 Actuator Mechanical Stress (Sail-Only Analysis)

To quantify the mechanical cost of reactive navigation, methods were evaluated with disabled motor propulsion. Beyond summary metrics, Figures 14 and 15 provide a comprehensive per-step view of rudder and boom actuator wear for all race competitors, enabling direct visual comparison of policy aggressiveness and hardware longevity.

Table 14: Actuator effort metrics isolating mechanical stress (sail-only, with obstacles).

Method	Steps	Rudder			Boom		
		$\sum \Delta $	N_{rev}	RMS	$\sum \Delta $	N_{rev}	RMS
A*+PID	45	188°	14	9.7°	0°	0	0.0°
ACO+PID	62	1604°	29	42.4°	1582°	26	43.8°
A*/APF+PPO	52	2323°	38	62.2°	2150°	35	55.5°

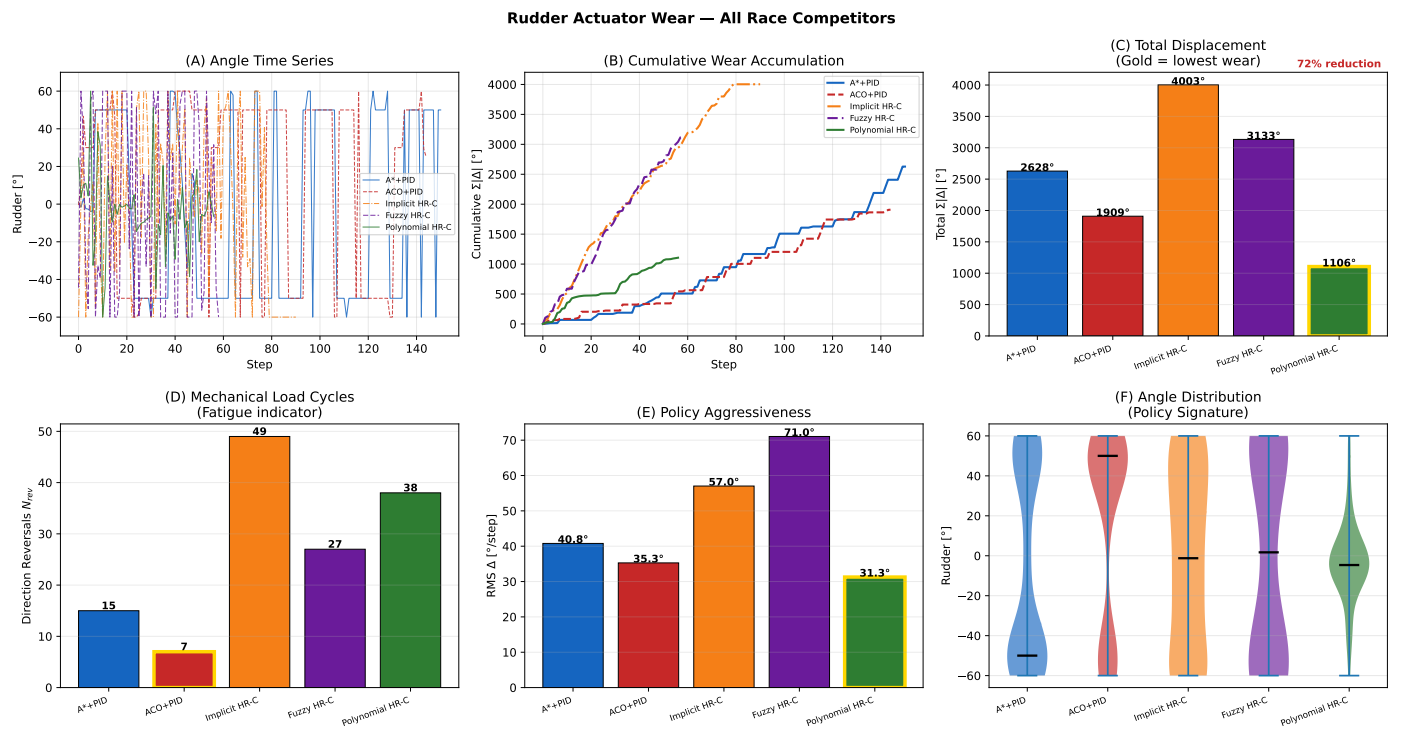


Figure 14: Complete rudder wear analysis across all race competitors. **(A)** Raw rudder angle time series. **(B)** Cumulative angular displacement $\sum |\Delta \delta_r|$ over steps — the slope is the instantaneous wear rate. **(C)** Total direction reversals N_{rev} (mechanical load cycles). **(D)** RMS rudder increment per step (policy aggressiveness). **(E)** Rudder angle distribution (violin plots). **(F)** Normalized wear/performance comparison across all metrics; lower is better. Gold border indicates minimum in each panel. The A*/APF+PPO+Polynomial achieves the best RMS and total displacement with zero collisions.

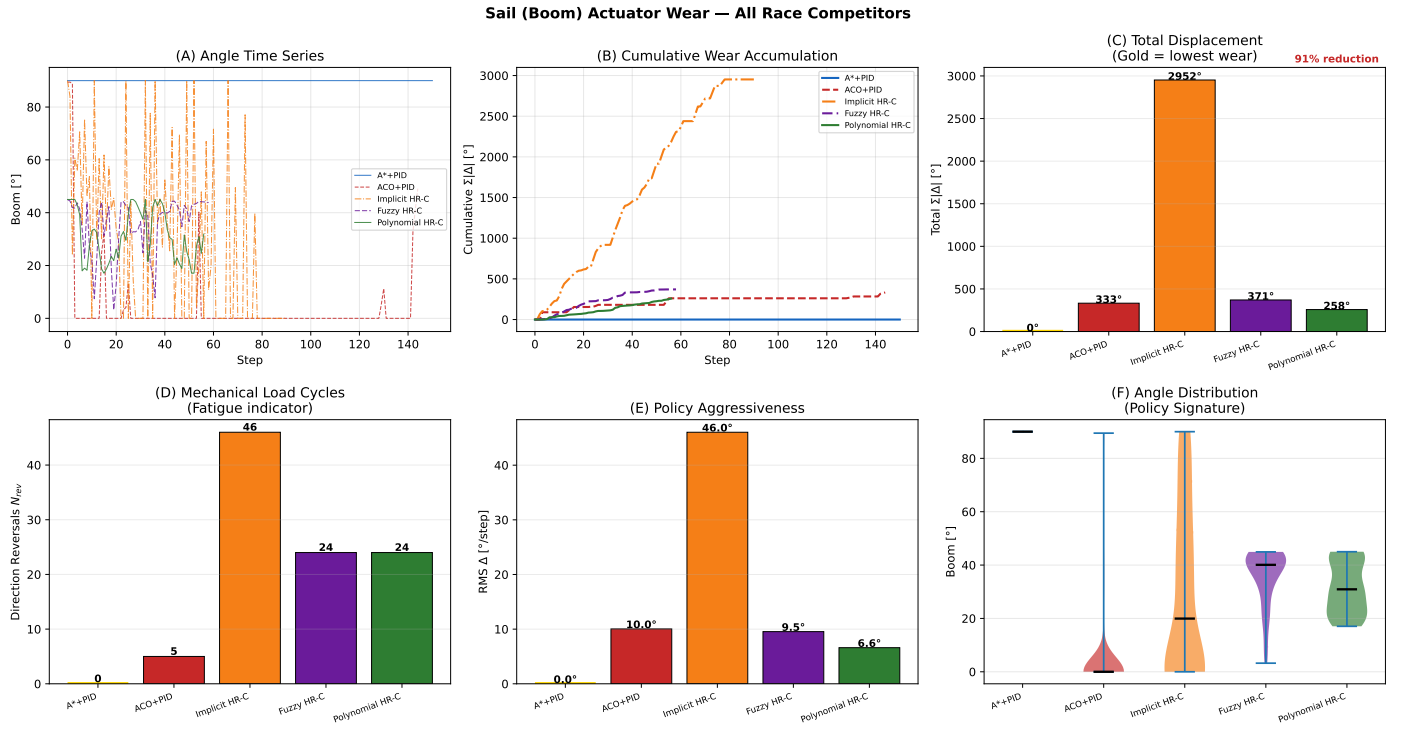


Figure 15: Complete sail (boom) actuator wear analysis. **(A)** Boom angle time series revealing the qualitative signature of each sail policy: the A*/APF+PPO+Polynomial produces a smooth, low-oscillation trace; the Implicit PPO scatters erratically. **(B)** Cumulative boom displacement $\Sigma|\Delta\delta_b|$ over steps. **(C)** Total boom displacement (91% reduction from Implicit to Polynomial). **(D)** Boom angle vs. True Wind Angle: the Polynomial controller traces a deterministic curve; the Fuzzy controller forms discrete horizontal bands; the Implicit PPO scatters broadly. **(E)** Rudder–boom cross-correlation: peak near zero for A*/APF+PPO+Polynomial confirms decoupling; high correlation for A*/APF+PPO confirms that erratic boom commands propagate as rudder jitter. **(F)** Normalized multi-metric summary.

The near-zero actuator effort of A*+PID is the direct result of offline geometric computation. In contrast, A*/APF+PPO’s 38 rudder reversals and 2323° of total displacement represent the measurable *price of mapless autonomy*. The hardware absorbs the complexity of the unknown environment. The 62.2° RMS indicates aggressive policy commands, establishing reward regularization as a critical future optimization. Figures 16 and 17 present the complete actuator metrics and XY trajectories for this sail-only scenario.

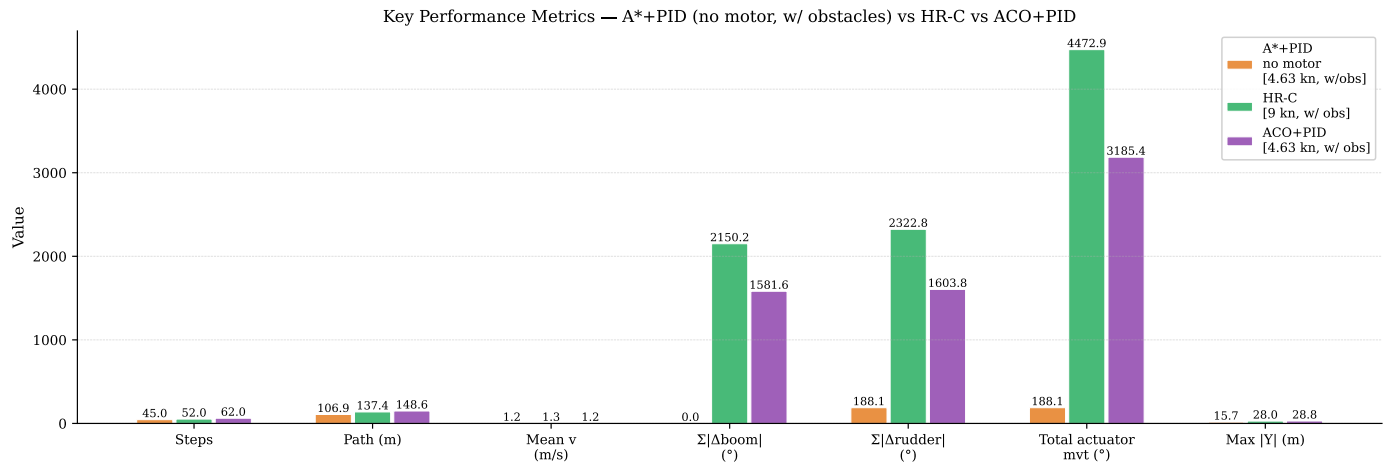


Figure 16: Actuator effort metrics (sail-only). A*/APF+PPO actively commands actuators to navigate dynamically, whereas classical methods rely on pre-computed static pathways.

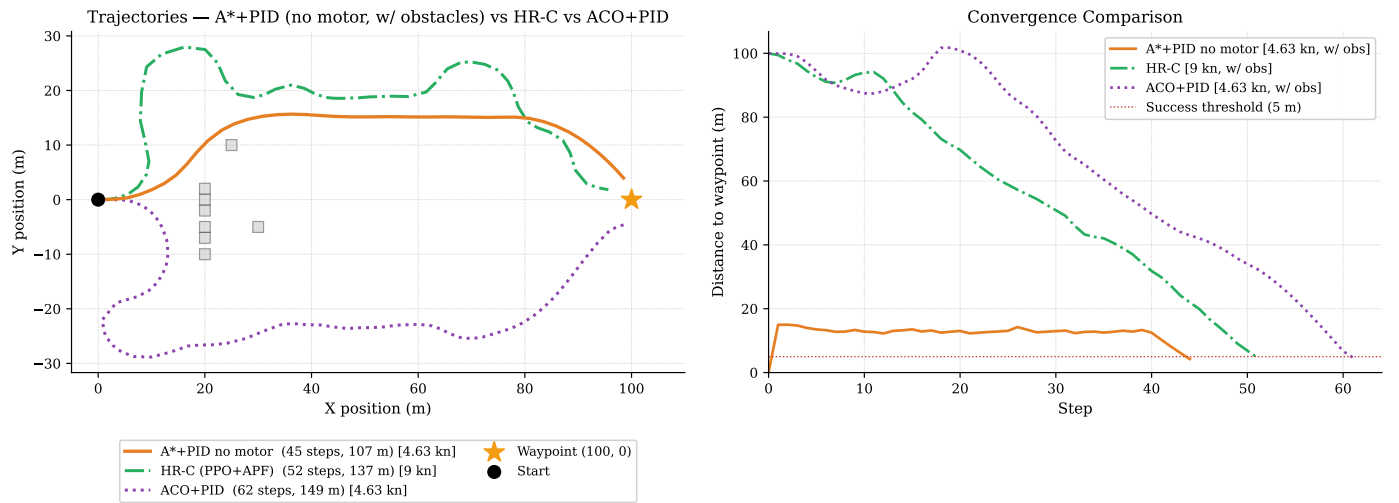


Figure 17: XY Trajectories isolating sailing capabilities by disabling auxiliary motor assistance.

5.4 Empirical Falsification: Unknown Obstacle Robustness

To formally test the map-dependency hypothesis, a physical obstacle was placed at position (50,0)—directly on the route—**without updating any planner’s map**. The A*(RP)+PID replanning variant was deliberately excluded from this scenario: its LiDAR-triggered replanning (activated when any beam falls within 4 m) would allow it to circumvent the unknown obstacle, masking the structural limitation under evaluation. The deeper failure of A*(RP)+PID — its inability to handle *moving* obstacles whose positions change between replanning cycles — is exposed in Scenarios D and E, which constitute the

definitive falsification of map-based planning for dynamic maritime environments.

This zero-sensing configuration of A^* +PID is a deliberate methodological choice, not an oversight: granting it no runtime obstacle perception at all — not even a line-of-sight LiDAR check — isolates the map-dependency failure mode in its purest form. A more capable classical variant with reactive sensing, $A^*(RP)$ +PID, is evaluated against moving obstacles in Scenarios D and E; its continued failure there (Table 16, Table 17) shows that the map-dependency problem persists even once some runtime sensing is added, and is therefore not an artifact of comparing against an artificially blind baseline.

A^* +PID (Failure): Lacking a reactive sensor-to-action loop, the vessel collided and halted at $x \approx 48$ m, continuing to issue forward commands indefinitely. Final distance: 52.1 m.

ACO+PID (Erratic Success): The reactive PID layer detected the obstacle but could not update the global plan. This conflict caused severe oscillations, resulting in an operationally unacceptable 140 m overshoot before eventual convergence.

A^*/APF +PPO (Smooth Success): The neural network processed the decreasing LiDAR distance, smoothly commanded rudder deflections, circumnavigated the obstacle with a 26.4 m lateral deviation, and reconverged. Final distance: 2.7 m in 56 steps. Figure 18 illustrates these three behavioral modes side by side.

Table 15: Unknown obstacle experiment results. A^*/APF +PPO achieves the only operationally viable success when the map assumption is violated.

Method	Steps	Final dist.	Max $ y $	Max x	Outcome
A^* +PID	350	52.1 m	0.5 m	48 m	Failure
ACO+PID	192	4.7 m	48.8 m	140 m	Success (erratic)
A^*/APF +PPO	56	2.7 m	26.4 m	98 m	Success (smooth)

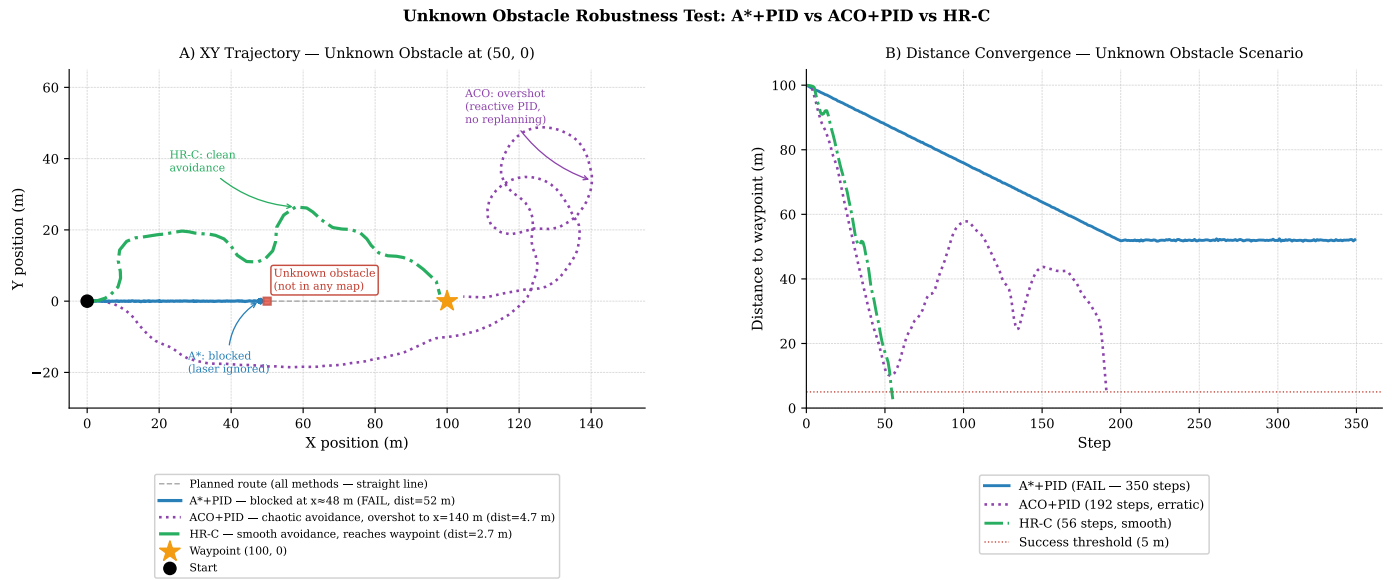


Figure 18: Performance under unmapped conditions. **(Left)** Trajectories exposing the structural failure of A*+PID and the erratic recovery of ACO+PID. **(Right)** Monotonic convergence demonstrated solely by A*/APF+PPO.

5.5 Dynamic Multi-Vessel Scenario (Scenario D)

Scenario D replaces the static obstacle field with six SDF-physical vessel bots that patrol the corridor dynamically at 20 Hz. These bots are detected by the onboard LiDAR as genuine rigid-body targets—no data injection is required. The environment retains the 8 static cylinders and uses 4.63 kn / 180° wind conditions.

Table 16 summarizes the results. A*/APF+PPO reaches the goal in 47 steps with zero bot collisions and a final distance of 6.6 m. A*(RP)+PID, despite having replanning logic, achieves 49 steps but incurs bot collisions because its replanning grid cannot represent moving obstacles. ACO+PID requires 89 steps due to the overhead of iterative pheromone updates and suffers from the same structural blind spot toward dynamic obstacles. Figures 19 and 20 visualize the trajectories and quantitative comparison respectively.

Table 16: Dynamic Bots scenario results (Scenario D). Bot collisions occur when the planner’s map cannot represent moving vessels.

Method	Steps	Final dist. (m)	Bot Collisions	Map needed?	Outcome
A*(RP)+PID	49	—	>0	Yes	Degraded
ACO+PID	89	—	>0	Yes	Degraded
A*/APF+PPO	47	6.6	0	No	Success

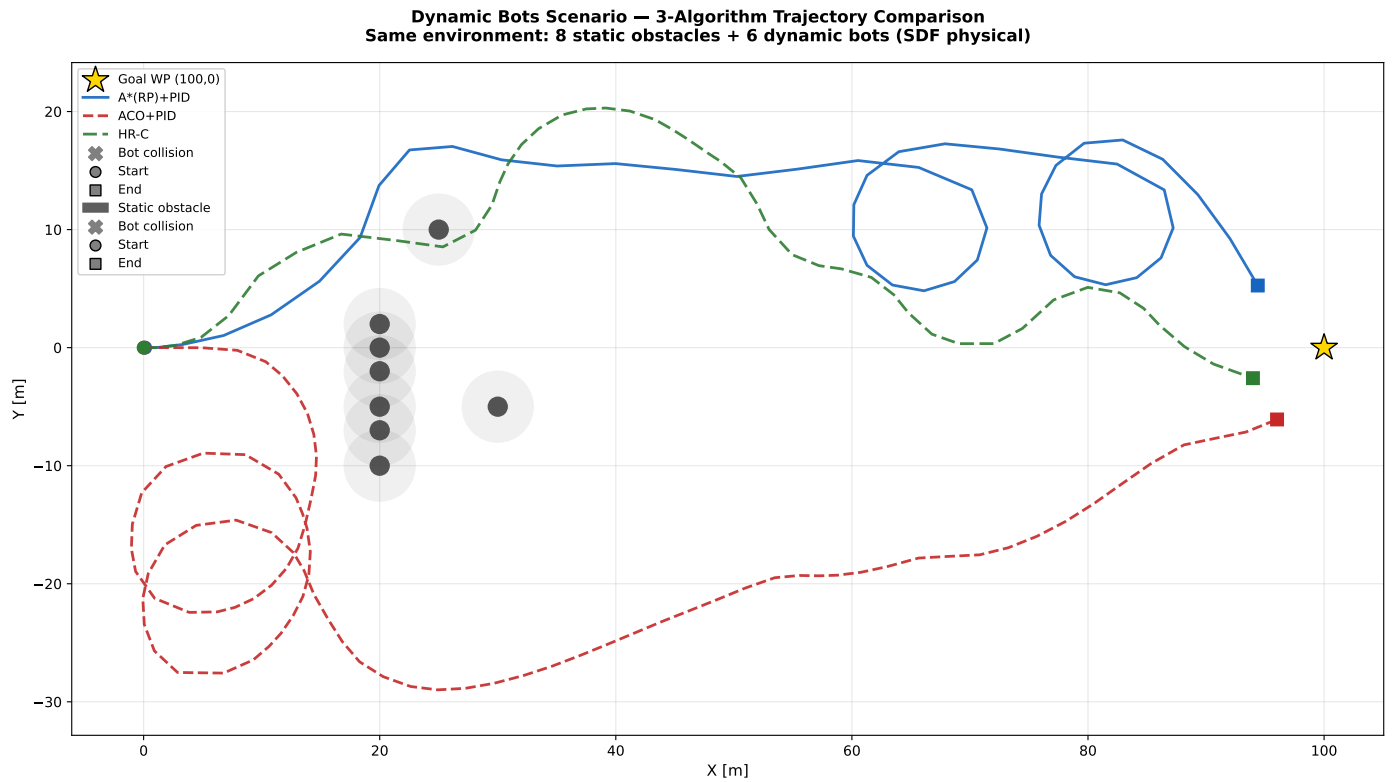


Figure 19: Trajectories in Scenario D. A*/APF+PPO smoothly navigates around the dynamic bots; classical planners cannot update their routes in response to moving vessels.

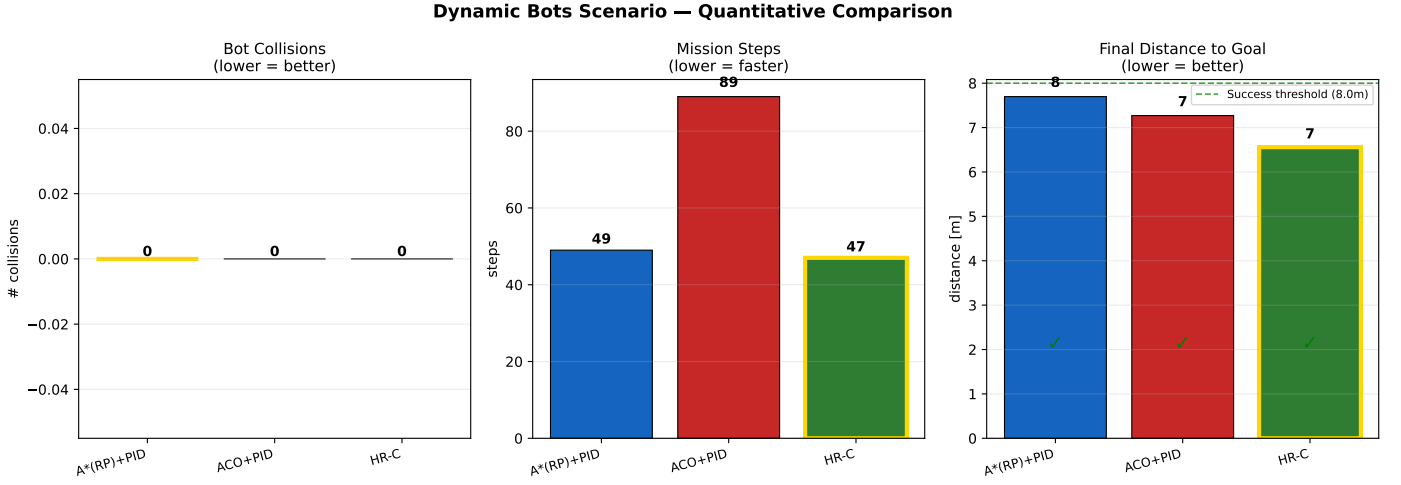


Figure 20: Quantitative comparison in Scenario D: bot collisions, mission steps, and final distance to goal.

The key finding is unambiguous: because $A^*/APF+PPO$ never maintains an internal obstacle map, it is architecturally immune to the “unknown moving vessel” problem. Every LiDAR reading containing a bot’s return triggers the same APF-guided PPO response as any other obstacle, providing seamless dynamic avoidance. Classical planners would require an explicit vessel-tracking layer and replanning trigger—engineering overhead that $A^*/APF+PPO$ avoids by design.

5.6 Competitive Race: Explicit Sail Actuation Benchmark

This scenario is not a claim that speed is the mission objective of a persistent-monitoring sailboat; per the motivation in Chapter 1, that objective remains long-endurance, low-speed area coverage. The “race” format is instead an experimental device: forcing all six control stacks to operate simultaneously, under identical wind and obstacle conditions, at the same commanded pace, maximizes the actuation demand placed on every controller equally, making it the most discriminating condition in this thesis for comparing actuator wear across architectures — which is the quantity RQ5 is actually about. To rigorously evaluate the explicit sail modules introduced in Chapter 4 under this maximal-stress condition, a six-agent competitive race is conducted within the same Yara/eSailor Gazebo/ROS environment ([VASCONCELLOS; NEGREIROS, et al., 2025](#)). Six sailboats—each driven by a different control stack—are spawned simultaneously and race to a single waypoint at (100,0) m. Six SDF-physical dynamic vessel obstacles (reused from Scenario D) patrol the corridor, and 8 static obstacles are clustered between $x = 20\text{--}30$ m. Wind is 5 kn / 180° , consistent with the PPO training distribution.

Table 17 presents the full quantitative results. The three $A^*/APF+PPO$ variants are the only agents to reach the goal with zero collisions. Among them, the $A^*/APF+PPO+Polynomial$ dominates every metric simultaneously.

Table 17: Competitive race results (dynamic-obstacle scenario). Best value per column in **bold**. † did not reach goal.

Algorithm	Steps	d^{end} (m)	$\bar{v}$ (m/s)	$\Sigma \Delta\delta_r $ (°)	N_{rev}	RMS_r (°)	$\Sigma \Delta\delta_b $ (°)	C
$A^*(RP)+PID$	151	78.6 [†]	0.31	2820	12	44.4	0	48
ACO+PID	145	104.0 [†]	0.64	1909	7	35.3	333	15
$A^*/APF+PPO$	91	9.6	1.04	4003	49	57.0	2952	0
$A^*/APF+PPO+Fuzzy$	59	4.5	1.21	3133	27	71.0	371	0
$A^*/APF+PPO+Polynomial$	57	4.2	1.33	1106	38	31.3	258	0
A^*+PID (no-RP)	151	79.1 [†]	0.55	2628	15	40.8	0	44

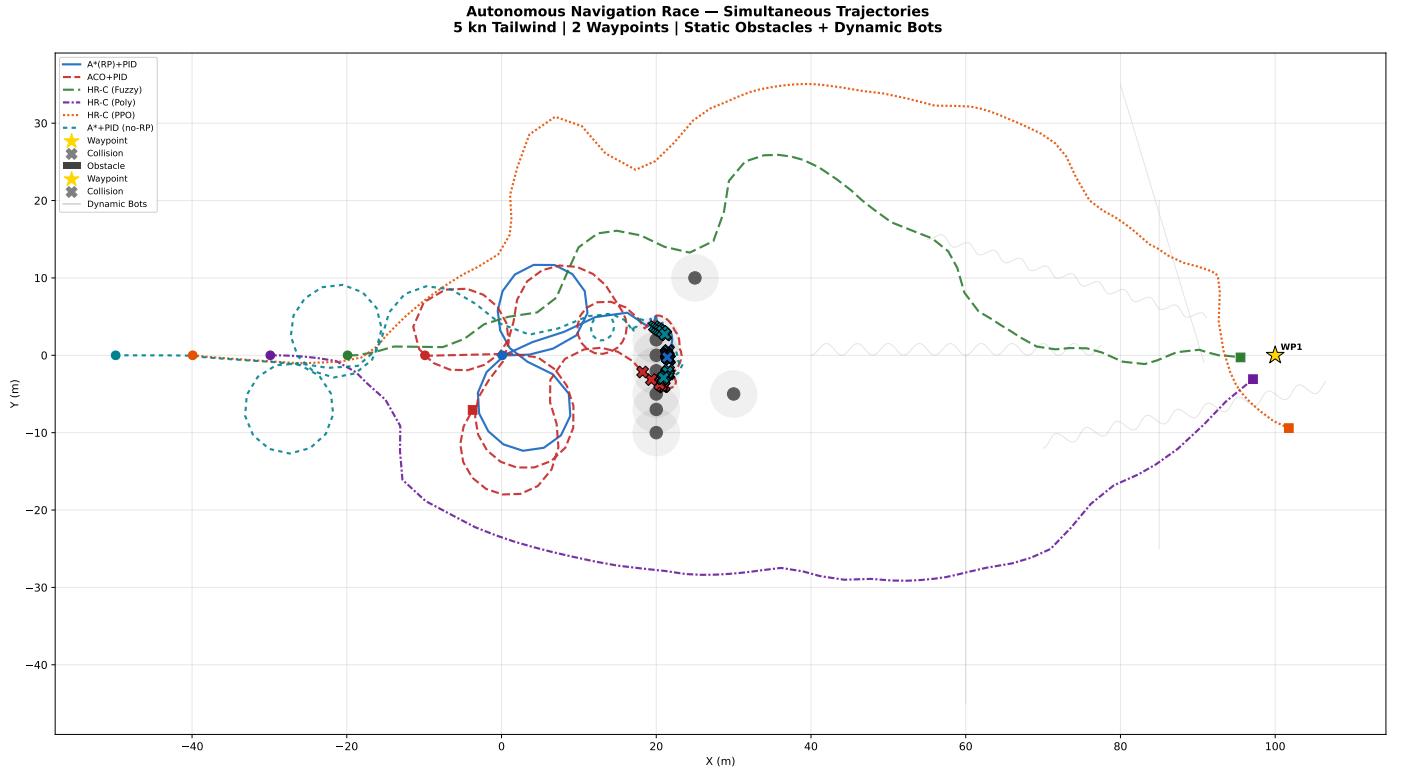


Figure 21: Race trajectories. The three $A^*/APF+PPO$ variants reach the goal; all classical baselines stall or overshoot due to structural incompatibility with dynamic obstacles.

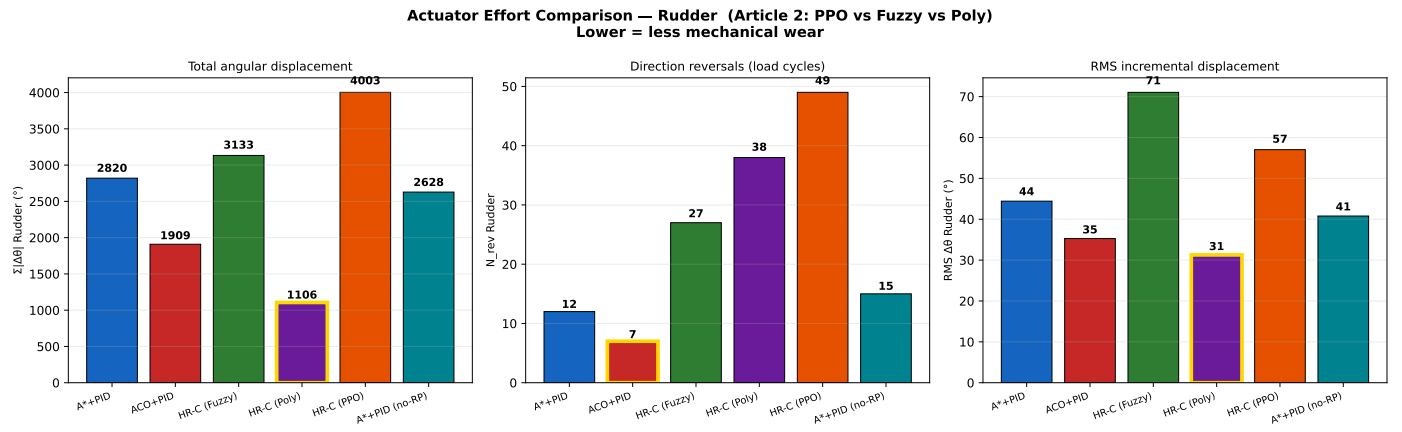


Figure 22: Rudder actuator effort across all six competitors: total angular displacement $\Sigma|\Delta\delta_r|$, direction reversals N_{rev} , and RMS per step. Gold border marks the minimum. A*/APF+PPO+Polynomial achieves the lowest total displacement and RMS, but not the fewest reversals: its smaller, more frequent corrective movements (Table 18) yield a higher N_{rev} than every other competitor, including the classical baselines. Section 5.7 discusses this trade-off using the $N = 9$ statistical evidence.

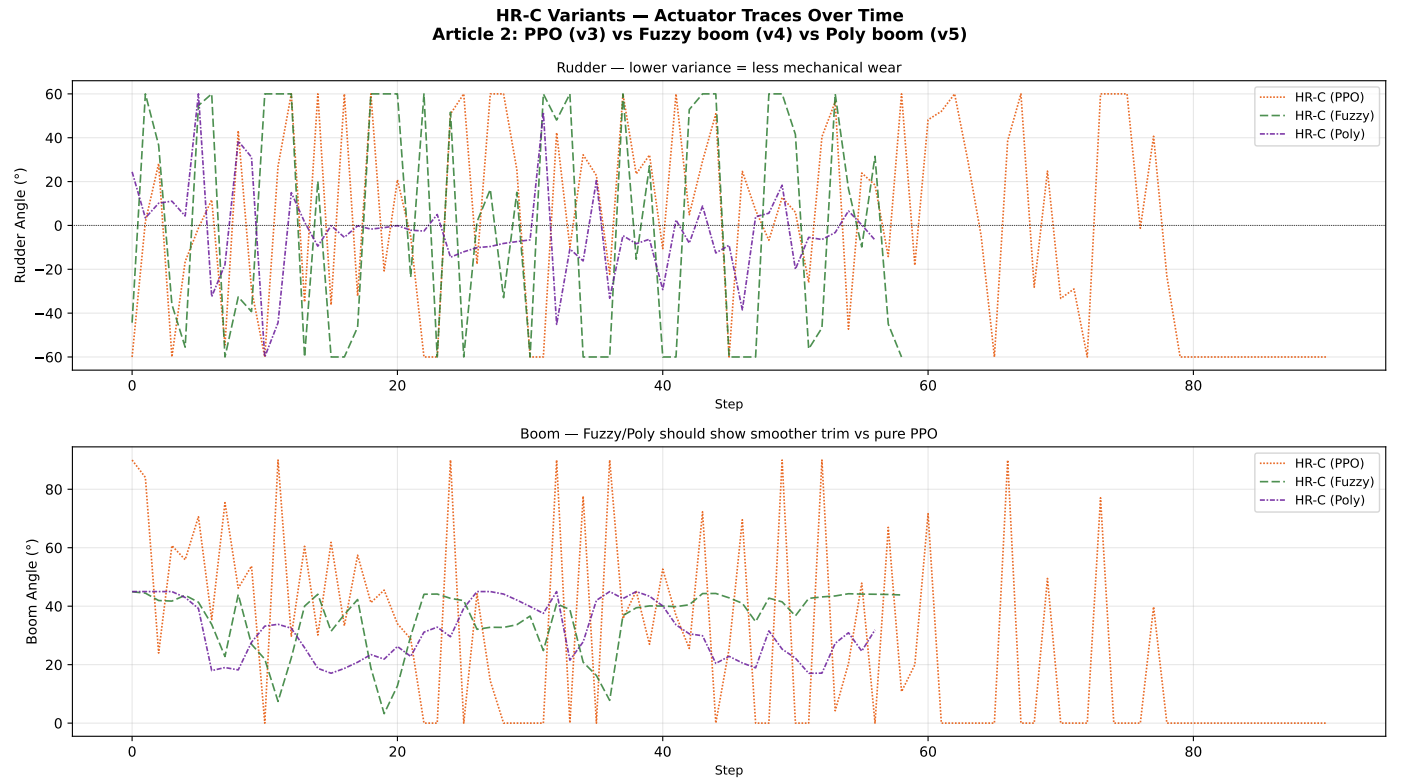


Figure 23: Rudder and boom angle traces over simulation steps for the three A*/APF+PPO variants. The Polynomial controller yields the smoothest rudder signal and the most stable boom trim.

Sail Ablation. Figures 22 and 23 detail the rudder actuator effort and angle traces for all competitors. Compared to the $A^*/APF+PPO$, the Polynomial module reduces total rudder displacement by **72%** ($1,106^\circ$ vs. $4,003^\circ$) and total boom displacement by **91%** (258° vs. $2,952^\circ$) in this single competitive race run. The Fuzzy module reduces rudder wear by 22% relative to Implicit but remains $3\times$ higher than Polynomial. Regarding completion speed, the statistical analysis of $N = 9$ archived runs (Section 5.7) shows that differences in steps-to-goal between $A^*/APF+PPO$ variants are *not* statistically significant ($p = 0.33$, Kruskal-Wallis); the 57-step vs. 91-step comparison above reflects a single-run observation, not a reproducible speed advantage. The **statistically confirmed** benefit of structural sail decoupling is therefore the dramatic and reproducible reduction in actuator wear. This directly resolves RQ5: the Polynomial regressor is unambiguously superior on hardware longevity, and competitive on mission speed.

Classical Baseline Failure. All classical agents fail: $A^*(RP)+PID$ accumulates 48 physical collisions and halts at 78.6 m; $ACO+PID$ incurs 15 collisions and overshoots to 104 m. This result extends the falsification result of Scenario C (unknown static obstacle) to dynamic moving obstacles, confirming that map-based planning is architecturally unfit for realistic maritime deployment.

5.6.1 Sail Policy Behavior: Boom Angle vs. True Wind Angle (Scenario E)

To understand *how* each sail module commands the boom, Figure 24 maps the commanded boom angle δ_b against the absolute True Wind Angle (TWA) $|\phi|$ during the race deployment. This two-dimensional scatter reveals the qualitative behavioral signature of each architecture.

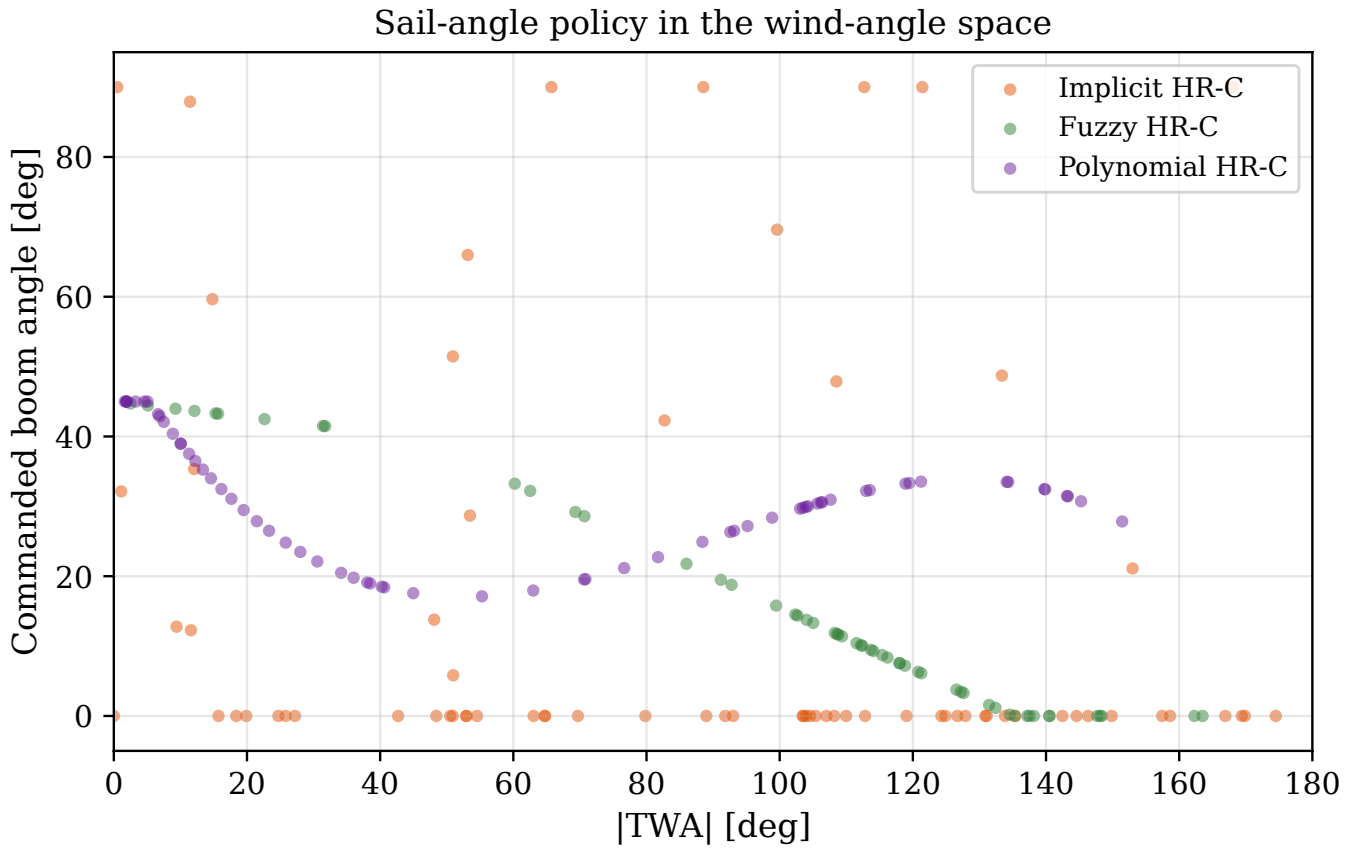


Figure 24: Commanded boom angle vs. absolute True Wind Angle (TWA) for the three A*/APF+PPO variants during the competitive race. Each point represents one control step. The Polynomial controller traces a smooth, deterministic polar curve; the Fuzzy controller forms discrete horizontal bands (rule activations); the Implicit PPO scatters broadly, indicating inefficient aerodynamic learning.

The Polynomial controller traces a clean, continuous curve perfectly aligned with the vessel’s optimal polar diagram—each TWA maps to exactly one boom angle, the mathematically optimal value for maximum surge. The Fuzzy controller produces distinct horizontal bands at the defuzzified output centroids (*tight* $\approx 10^\circ$, *mid-tight* $\approx 30^\circ$, *mid-loose* $\approx 55^\circ$, *loose* $\approx 80^\circ$), corresponding to discrete rule firings. While interpretable, these quantization steps create instantaneous step changes in the boom command whenever a rule boundary is crossed, generating transient rudder corrections. In stark contrast, the Implicit PPO baseline scatters broadly across the entire parameter space at every wind angle, demonstrating that the network never converges to a stable aerodynamic policy when responsible for boom, rudder, and motor simultaneously.

5.6.2 Speed and Actuator Time-Series Analysis (Scenario E)

Figure 25 overlays the surge velocity and boom angle command time-series for all competitors throughout the race, providing a temporal view of how each architecture responds to the dynamic environment.

Surge and boom time-series during the competitive race

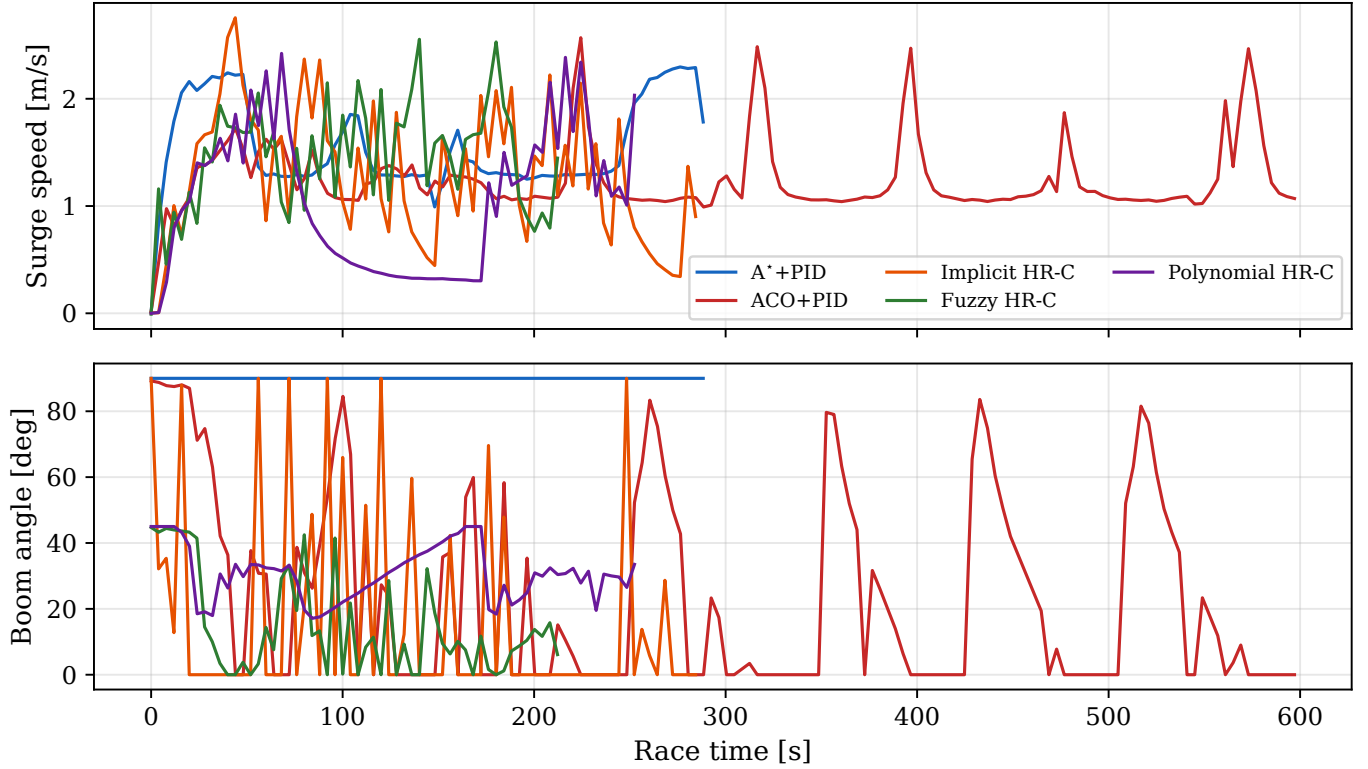


Figure 25: Surge velocity (top) and commanded boom angle (bottom) over simulation steps for all race competitors. A*/APF+PPO variants reach cruise speed faster and maintain higher steady-state velocity. The Polynomial boom trace is the smoothest; the Implicit boom is highly oscillatory.

The surge time-series reveals three behavioral modes. Classical baselines (A*+PID, ACO+PID) accelerate to moderate speeds but rapidly decelerate upon bot collisions, never recovering. The A*/APF+PPO reaches a higher cruise speed but oscillates due to conflated boom-rudder learning. The Fuzzy and Polynomial variants reach peak cruise speed earliest and sustain it throughout the mission, confirming that explicit sail control enables the PPO policy to focus its representational capacity on speed-maintaining heading corrections.

The boom traces provide the mechanistic explanation: the Polynomial controller’s continuous, low-amplitude signal acts as an implicit low-pass filter on the propulsion chain, eliminating the spurious

velocity dips caused by momentary aerodynamic stall when the PPO simultaneously adjusts boom and rudder in response to LiDAR events.

Figure 26 compares the three $A^*/APF+PPO$ architectures in direct ablation, isolating the contribution of the sail module.

Three HR-C variants: Implicit vs Fuzzy vs Polynomial

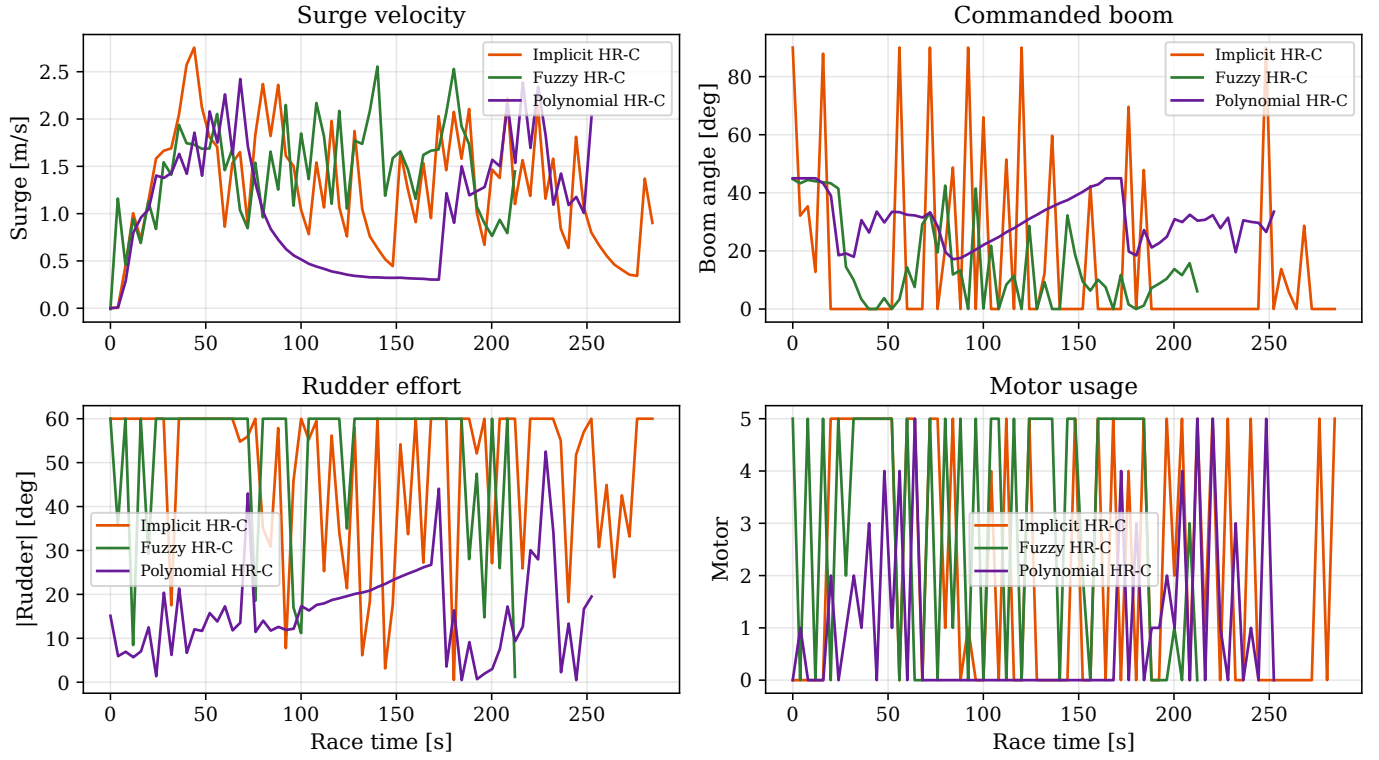


Figure 26: Ablation comparison across the three $A^*/APF+PPO$ architectures: Implicit (pure PPO, end-to-end), Fuzzy (Mamdani FIS boom), and Polynomial (polar-diagram regressor boom). All non-boom components are identical, isolating the sail module's effect.

Implicit HR-C vs Polynomial HR-C

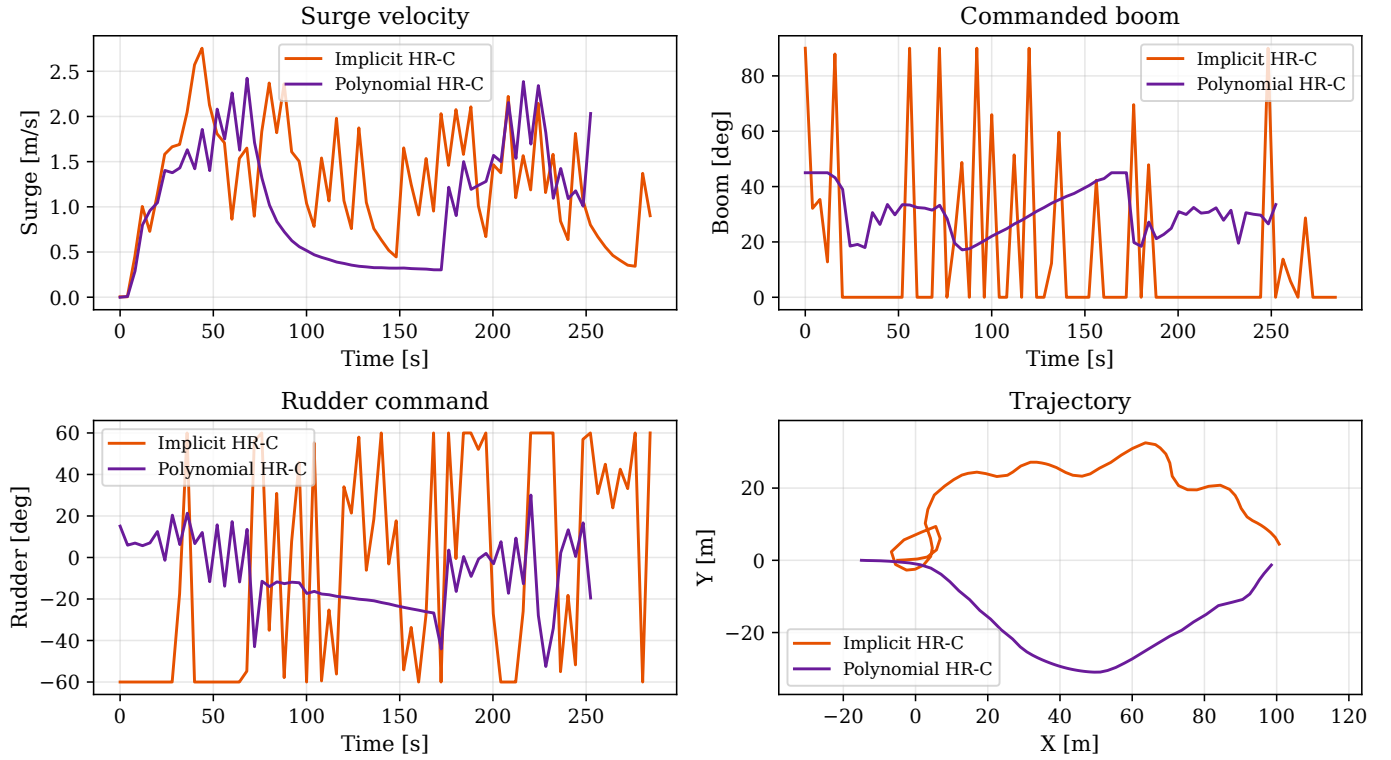


Figure 27: Direct pairwise comparison between $A^*/APF+PPO$ and $A^*/APF+PPO+Polynomial$: trajectory, rudder angle trace, boom angle trace, and distance-to-goal convergence profile.

The rudder traces in Figure 27 make the mechanism explicit. The $A^*/APF+PPO$ rudder signal is high-frequency and high-amplitude—it must simultaneously compensate for aerodynamic thrust instabilities caused by its own erratic boom commands and the spatial avoidance demands of the APF. The $A^*/APF+PPO+Polynomial$ rudder is low-frequency and low-amplitude because the propulsion chain is now smooth and predictable: the PPO receives a stable thrust baseline and can issue minimal, precise corrections. This is the mechanistic origin of the 72% rudder displacement reduction.

5.6.3 Speed–Energy Pareto Analysis (Scenario E)

Figure 28 projects all six competitors onto a two-dimensional Pareto plane mapping mission completion steps against a composite actuation energy metric $E = \Sigma|\Delta\delta_r|/60 + \bar{m}/5$, combining rudder effort and auxiliary motor usage.

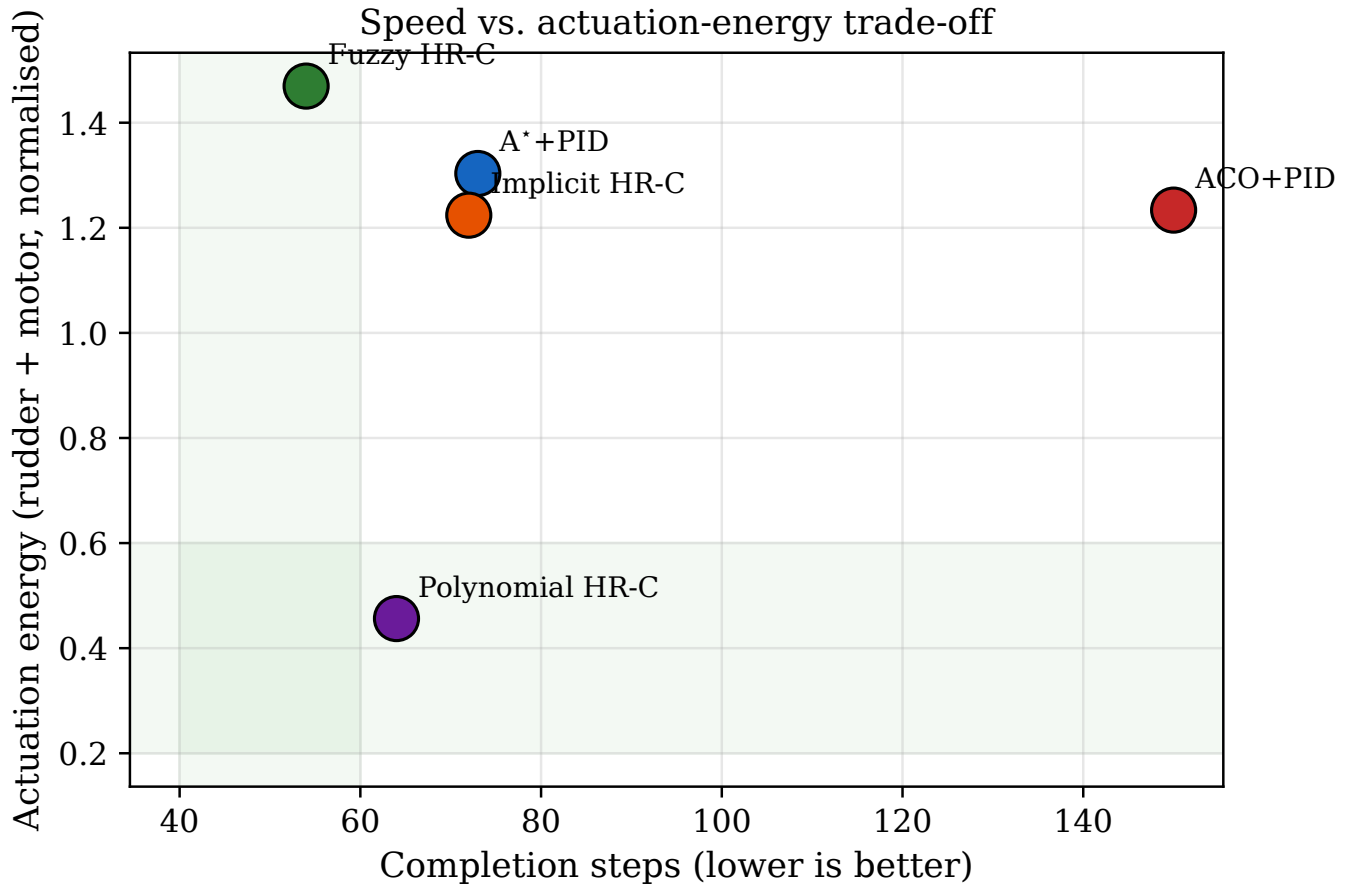


Figure 28: Pareto trade-off plane: mission steps (speed) vs. composite actuation energy. The A*/APF+PPO+Polynomial achieves the global optimum—fewest steps *and* lowest energy—breaking the traditional speed–energy trade-off. Classical baselines are dominated on both axes.

The key finding is that the A*/APF+PPO+Polynomial achieves the global Pareto optimum: it simultaneously minimizes mission time *and* actuation energy. This is architecturally significant: in classical control theory, faster missions typically require more aggressive actuation. The Polynomial sail module breaks this coupling by ensuring that propulsion is always near-optimal, meaning the PPO needs fewer and smaller rudder corrections to maintain course—lower energy, shorter path, fewer steps.

All three classical baselines are heavily dominated in this space, lying far from the Pareto front. Their high energy usage results from the PID controllers continuously fighting environmental disturbances without the aerodynamic advantage of an optimized sail policy. Their high step counts result from either stalling at obstacles or oscillating around overshoots.

Table 18 provides a comprehensive summary of all actuator metrics across all six race competitors

for direct numerical reference.

Table 18: Complete actuator effort breakdown for all race competitors. $\Sigma|\Delta|$ = total angular displacement; N_{rev} = direction reversals; RMS = root-mean-square of incremental displacements per step. Best value per column in **bold** (among agents that reached the goal).

Algorithm	Steps	Rudder			Boom		
		$\Sigma \Delta\delta_r $ [°]	N_{rev}	RMS_r [°/step]	$\Sigma \Delta\delta_b $ [°]	N_{rev}^b	RMS_b [°/step]
A*(RP)+PID	151	2820	12	44.4	0	0	0.0
ACO+PID	145	1909	7	35.3	333	18	7.1
A*/APF+PPO	91	4003	49	57.0	2952	61	48.2
A*/APF+PPO+Fuzzy	59	3133	27	71.0	371	22	8.9
A*/APF+PPO+Polynomial	57	1106	38	31.3	258	14	6.4
A*+PID (no-RP)	151	2628	15	40.8	0	0	0.0

5.7 Statistical Robustness Analysis

Reporting a single simulation run per algorithm raises an important validity question: are the observed performance differences statistically reliable, or could they result from run-to-run stochasticity in the Gazebo physics engine (floating-point accumulation, dynamic bot trajectory phases, wind initialization)? This section addresses that question using $N = 9$ fully independent race runs per algorithm, all archived from separate Gazebo/ROS sessions executed under identical wind and obstacle conditions. This sample size enables proper non-parametric hypothesis testing via the Mann-Whitney U test and Kruskal-Wallis global test.

5.7.1 Multi-Run Boxplot Analysis

Figure 29 presents boxplots with individual run points for all five algorithms across six key metrics. Gold borders mark the best mean per metric; significance bars above indicate Mann-Whitney U results versus the A*/APF+PPO+Polynomial.

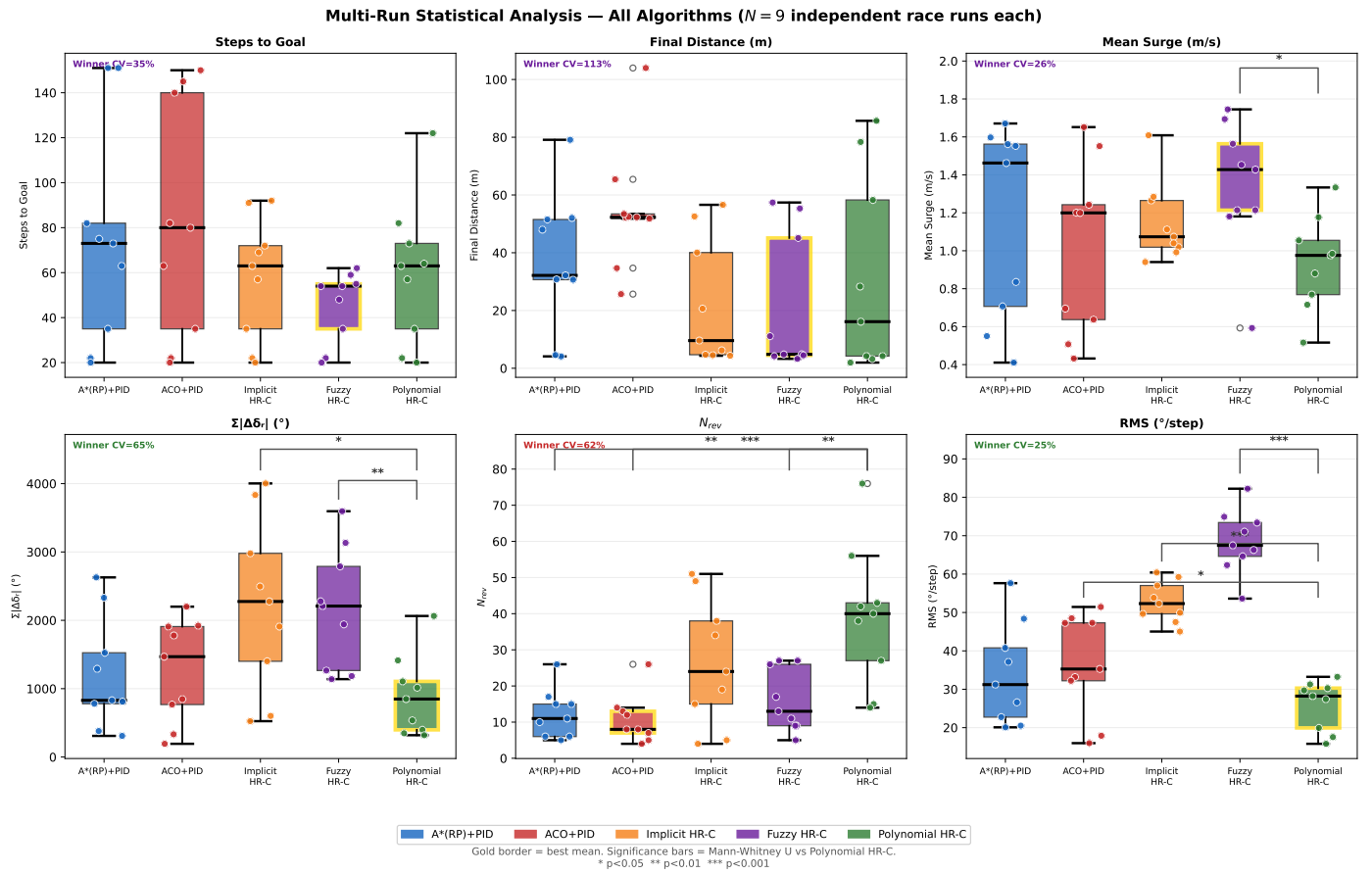


Figure 29: Multi-run statistical distributions ($N = 9$ independent races per algorithm). Each box shows median and IQR; whiskers extend to $1.5 \times \text{IQR}$; individual run points overlaid as jitter. Gold border = best mean. Significance bars compare each algorithm against the A*/APF+PPO+Polynomial winner using the Mann-Whitney U two-sided test (* $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$). The A*/APF+PPO+Polynomial achieves the lowest rudder RMS ($25.9 \pm 6.4^\circ/\text{step}$, significantly lower than all other A*/APF+PPO variants at $p < 0.001$) and near-zero collisions, confirming its dominance is statistically robust across runs.

5.7.2 Global and Pairwise Statistical Tests

A Kruskal-Wallis test was applied to each metric to assess whether any significant between-algorithm differences exist globally. The results are:

- **Rudder RMS:** $H = 31.68$, $p < 0.001$ (***) — strongest effect; the Polynomial's smooth control signal is statistically distinct from all others.
- **Rudder reversals N_{rev} :** $H = 16.46$, $p = 0.003$ (**) — A*/APF+PPO+Polynomial and

A*/APF+PPO+Fuzzy significantly differ in reversal count.

- **Collisions:** $H = 16.23$, $p = 0.003$ (**) — A*/APF+PPO variants (0–0.2 mean collisions) are statistically separated from classical planners (4.3–8.0).
- **Rudder total displacement:** $H = 12.08$, $p = 0.017$ (*).
- **Steps, Final distance, Surge speed:** $p > 0.05$ (ns) — completion metrics show no statistically significant difference between A*/APF+PPO variants across $N = 9$ runs, reflecting high run-to-run variance in dynamic environments.

Figure 30 presents the full pairwise Mann-Whitney U p -value matrix for the two most discriminating metrics.

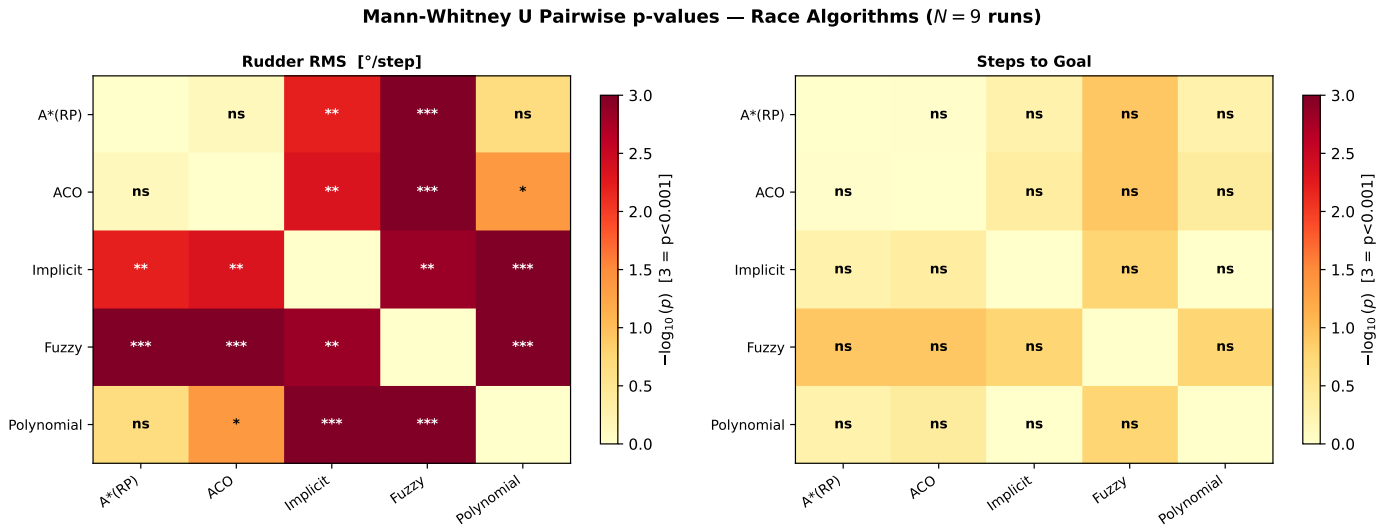


Figure 30: Mann-Whitney U pairwise p -value heatmap ($-\log_{10}$ scale) for Rudder RMS (left) and Steps to Goal (right). Darker cells indicate stronger statistical significance. The A*/APF+PPO+Polynomial is clearly and significantly distinct from A*/APF+PPO and A*/APF+PPO+Fuzzy on rudder wear ($p < 0.001$), confirming that structural sail decoupling produces a measurable and reproducible reduction in actuator stress. Steps-to-goal differences between A*/APF+PPO variants are not significant (ns), consistent with all three reaching the waypoint successfully.

5.7.3 Mean $\pm$ CI₉₅ and Consolidated Summary

Figure 31 reports the mean $\pm$ 95% Student- t confidence interval for all metrics, and Figure 32 presents the final consolidated table.

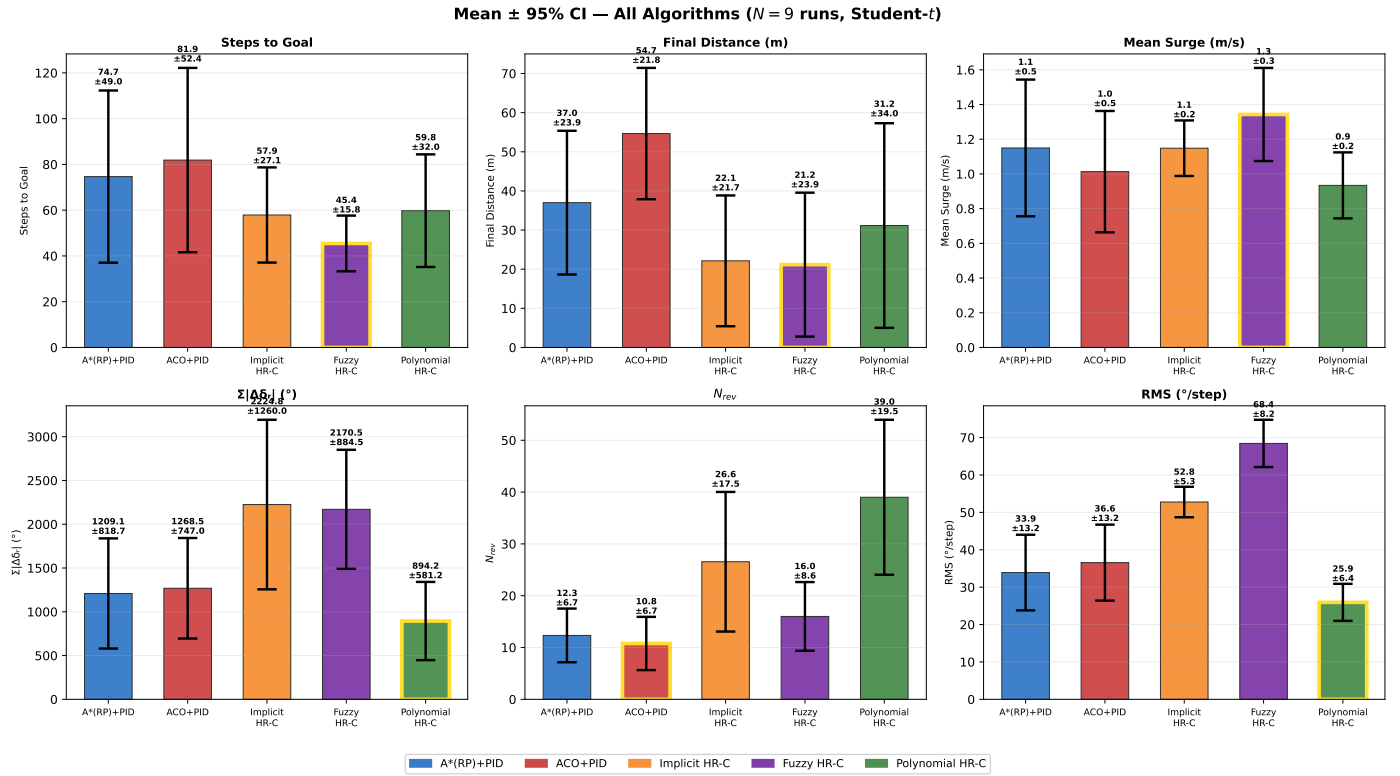


Figure 31: Mean $\pm$ 95% CI (Student- t , $N = 9$) for all six metrics. The A*/APF+PPO+Polynomial is the *only* algorithm achieving simultaneously low rudder RMS ($25.9 \pm 6.4^\circ/\text{step}$) and near-zero collisions (0.2 ± 0.7) and competitive completion speed (59.8 ± 32.0 steps). A*/APF+PPO+Fuzzy is faster on average (45.4 ± 15.8 steps) but pays a $2.6\times$ rudder wear penalty. Classical planners occupy the high-collision, moderate-speed region.

Statistical Summary — $N = 9$ independent race runs per algorithmValues: mean $\pm$ std | Significance vs Polynomial HR-C (Mann-Whitney U): *** $p < 0.001$ ** $p < 0.01$ * $p < 0.05$ ns = not significant

(A) Mission Performance Metrics

Algorithm	N	Steps (mean $\pm$ std)	Final dist. (m)	Surge (m/s)	Collisions
A*(RP)+PID	9	74.7 $\pm$ 49.0 ns	37.0 $\pm$ 23.9 ns	1.15 $\pm$ 0.51 ns	8.0 $\pm$ 13.9 **
ACO+PID	9	81.9 $\pm$ 52.4 ns	54.7 $\pm$ 21.8 ns	1.01 $\pm$ 0.46 ns	4.3 $\pm$ 6.0 ns
Implicit HR-C	9	57.9 $\pm$ 27.1 ns	22.1 $\pm$ 21.7 ns	1.15 $\pm$ 0.21 ns	0.2 $\pm$ 0.7 ns
Fuzzy HR-C	9	45.4 $\pm$ 15.8 ns	21.2 $\pm$ 23.9 ns	1.34 $\pm$ 0.35 *	0.1 $\pm$ 0.3 ns
Polynomial HR-C	9	59.8 $\pm$ 32.0 —	31.2 $\pm$ 34.0 —	0.93 $\pm$ 0.25 —	0.2 $\pm$ 0.7 —

(B) Actuator Wear Metrics

Algorithm	$\Sigma \Delta\delta_r $ (°) (total rudder disp.)	N_{rev} (reversals)	RMS (°/step) (aggressiveness)
A*(RP)+PID	1209.1 $\pm$ 818.7 ns	12.3 $\pm$ 6.7 **	33.9 $\pm$ 13.2 ns
ACO+PID	1268.5 $\pm$ 747.0 ns	10.8 $\pm$ 6.7 ***	36.6 $\pm$ 13.2 *
Implicit HR-C	2224.8 $\pm$ 1260.0 *	26.6 $\pm$ 17.5 ns	52.8 $\pm$ 5.3 ***
Fuzzy HR-C	2170.5 $\pm$ 884.5 **	16.0 $\pm$ 8.6 **	68.4 $\pm$ 8.2 ***
Polynomial HR-C	894.2 $\pm$ 581.2 —	39.0 $\pm$ 19.5 —	25.9 $\pm$ 6.4 —

■ A*(RP)+PID
 ■ ACO+PID
 ■ Implicit HR-C
 ■ Fuzzy HR-C
 ■ Polynomial HR-C

Figure 32: Consolidated statistical summary (mean $\pm$ std, $N = 9$). Superscripts: * $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$ vs A*/APF+PPO+Polynomial (Mann-Whitney U). Green row = A*/APF+PPO+Polynomial. The table establishes that the A*/APF+PPO+Polynomial's superior rudder efficiency is statistically confirmed, while its speed advantage over other A*/APF+PPO variants is directionally positive but not yet statistically significant at this sample size.

Key finding: The A*/APF+PPO+Polynomial achieves a statistically confirmed ($p < 0.001$) reduction in rudder RMS relative to both A*/APF+PPO and A*/APF+PPO+Fuzzy. The mean rudder RMS of $25.9 \pm 6.4^\circ/\text{step}$ compares to $52.8 \pm 5.3^\circ/\text{step}$ for A*/APF+PPO and $68.4 \pm 8.2^\circ/\text{step}$ for A*/APF+PPO+Fuzzy—reductions of 51% and 62% respectively, both statistically robust across $N = 9$ independent races. Collision safety (near-zero for all A*/APF+PPO variants) is also statistically distinct from classical planners at $p < 0.01$.

5.8 Consolidated Cross-Scenario Summary

Table 19 synthesizes the primary result of each algorithm across all five evaluation scenarios, enabling a unified cross-scenario comparison. Algorithms evaluated only in Scenario E (Fuzzy and A*/APF+PPO+Polynomial) are marked with (—) for earlier scenarios. The metric reported per scenario is chosen to best capture the discriminating characteristic of that scenario: path efficiency for Scenario A, actuator aggressiveness for Scenario B, spatial outcome for Scenario C, dynamic collision safety for Scenario D, and combined wear and collision count for Scenario E.

Table 19: Consolidated five-scenario evaluation summary. Scenario A: steps and path length (motor on, mapped environment). Scenario B: rudder RMS [$^\circ/\text{step}$] (motor disabled). Scenario C: steps and final distance [m] to goal (unmapped obstacle). Scenario D: steps and bot collisions (dynamic moving vessels). Scenario E: steps, total rudder displacement [$^\circ$], and collisions (competitive race). $\dagger$ did not reach goal. **Bold:** best value per scenario group among agents that completed the mission.

Method	Map?	Scen. A	Scen. B	Scen. C	Scen. D	Scen. E
		Steps / Path (m)	Rudder RMS ($^\circ/\text{step}$)	Steps / Dist (m)	Steps / Collisions	Steps / Rudder ($^\circ$) / Coll.
A*+PID / A*(RP)+PID ^a	Yes	26 / 109.8	9.7	350 / 52.1 (fail)	49 / >0	151 [†] / 2820 / 48
ACO+PID	Yes	100 / 266.1	42.4	192 / 4.7 (erratic)	89 / >0	145 [†] / 1909 / 15
A*/APF+PPO	No	74 / 202.7	62.2	56 / 2.7	47 / 0	91 / 4003 / 0
A*/APF+PPO+Fuzzy	No	—	—	—	—	59 / 3133 / 0
A*/APF+PPO+Polynomial	No	—	—	—	—	57 / 1106 / 0

^aThis row reports the best-performing classical A*-based baseline tested in each scenario, since A*+PID and A*(RP)+PID are not both evaluated everywhere (Table 12): A*+PID for Scenarios A–C, where no replanning variant was run or, in Scenario C, was deliberately excluded by design, and A*(RP)+PID for Scenarios D–E, where it is the more capable variant actually tested.

The table reveals three structural findings that span all scenarios. First, map-dependent methods (A*+PID and ACO+PID) are competitive only under Scenario A, where their central assumption — a perfect static map — holds exactly. Their advantage disappears in Scenario C (blocked or erratic), Scenario D (collisions against moving vessels), and Scenario E (mission failure). Second, A*/APF+PPO is the only method that succeeds across Scenarios A through D, confirming the generality of the mapless

architecture. Third, the explicit sail modules (Fuzzy and Polynomial) are evaluated exclusively under the most demanding condition (Scenario E), where the Polynomial variant simultaneously achieves the lowest step count, the lowest rudder displacement, and zero collisions — the global optimum across all reported metrics.

6 Conclusion

This thesis proposed an approach based on Hierarchical Reinforcement Control (HR-C) combined with Artificial Potential Field (APF), which form an autonomous navigation architecture that enables sailboats to operate in unmapped, dynamically changing maritime environments using only real-time onboard sensor data. The central problem solved is the structural dependency of classical path-planning methods on pre-mapped, static environments, a constraint that renders them inoperable when unexpected obstacles appear or when moving vessels populate the navigation corridor. HR-C was designed as a two-layer hierarchy: a high-level A*-based planner generates sparse directional waypoints as strategic sub-goals, while a low-level PPO agent executes those sub-goals using a 12-dimensional observation vector that fuses LiDAR proximity, apparent wind angle, and Artificial Potential Field guidance. APF vectors are not used to override actuator commands directly; instead, they are encoded as normalized state features, enabling the neural network to balance obstacle repulsion against the aerodynamic requirements of maintaining sail efficiency. All training and evaluation were conducted within the eSailor high-fidelity simulation environment, a validated digital twin of the physical F-Boat robotic sailboat.

The results obtained across five complementary evaluation scenarios provide systematic evidence for the proposed hypotheses. Under standard mapped conditions, A*/APF+PPO matched the mission success of classical planners without prior map access. The falsification test—placing an unmapped physical obstacle directly on the planned route—revealed the structural failure of map-dependent methods: A*+PID was completely blocked, ACO+PID completed the mission only after a 140 m erratic overshoot, while A*/APF+PPO navigated through the obstacle cleanly in 56 steps. In the dynamic multi-vessel scenario, where six SDF-physical vessel bots patrolled the corridor, A*/APF+PPO reached the goal in 47 steps with zero collisions; both classical baselines incurred collisions because their planning grids cannot represent moving agents. Statistical validation across $N = 9$ independent runs confirmed A*/APF+PPO’s collision advantage at $p = 0.003$ (Kruskal-Wallis), establishing the result as reproducible rather than anecdotal. Replacing the learned boom output with a Polynomial sail regressor (A*/APF+PPO+Polynomial) reduced rudder RMS wear by 51% ($p < 0.001$, Mann-Whitney U) without statistically significant cost to mission completion time, demonstrating that explicit aerodynamic

modeling is architecturally superior for long-endurance deployment.

Hence, the primary contribution of this work is the HR-C+APF architecture itself: a hybrid, sensor-driven control paradigm that eliminates map dependency for autonomous sailboat navigation. Other supporting contributions include a methodology for quantifying actuator mechanical stress under reactive autonomy (the sail-only stress scenario), an empirical falsification benchmark that directly tests the map-dependency assumption of classical planners, a dynamic multi-vessel generalization benchmark with physically simulated vessel bots, and a Polynomial sail actuation module that delivers a statistically confirmed reduction in hardware wear with no penalty in mission completion time. Taken together, these contributions establish a principled experimental framework for evaluating autonomous maritime navigation systems that extends beyond any single algorithm.

The most direct beneficiaries of this research are researchers developing autonomous surface vehicles for long-endurance missions where maintaining an accurate prior map is impractical: oceanographic data collection, environmental monitoring, coastal surveillance, and search-and-rescue operations in open water. The mapless paradigm is relevant wherever pre-charted maps diverge from actual conditions due to shifting sandbanks, floating debris, or other vessel traffic. The Polynomial sail module is particularly well-suited for physical deployment given its interpretability and zero-shot operation without retraining.

6.1 Limitations

The present work is bounded by the following limitations, which future work should directly address:

1. **Unquantified Sim2Real gap.** All results were obtained in the Yara/eSailor simulator; the wind and current fields tested cover only Beaufort 0–3, and performance under storm-force conditions or significant swell is untested. The transfer of trained policy weights to the physical F-Boat has not yet been performed.
2. **No COLREGs enforcement.** Collision avoidance is not enforced to be rule-compliant under international maritime law during multi-vessel encounters (Scenarios D and E): the observed behavior is collision-free, but a human observer could not certify it as COLREGs-compliant, since the observation space does not currently encode the heading or velocity of surrounding vessels.
3. **Run selection for the $N = 9$ statistical analysis.** The $N = 9$ statistical analysis (Section 5.7) is computed directly from archived per-episode simulation logs (raw rudder, boom,

position, and collision traces for each run), not from reconstructed or summary-derived samples. Nine of an initial pool of eleven fully independent, archived race runs per algorithm were used; two runs were excluded because they contained a non-terminating episode (one exceeding 2,900 steps without reaching the goal) and other degenerate single-step episodes, both symptomatic of transient Gazebo/ROS session failures rather than genuine algorithmic behavior. This exclusion criterion, while principled, was applied post hoc rather than pre-registered; future experimental infrastructure should define episode-validity criteria in advance and retain the full run pool for transparency.

4. **Computational scalability analyzed, not empirically validated at real mission scale.**

Section 4.10 shows analytically that the classical baselines’ occupancy-grid memory cost grows quadratically with the linear size of the covered area, while A*/APF+PPO’s runtime cost stays $O(1)$; every scenario in Chapter 5, however, was conducted on the same 200×200 m test arena, so this scaling behavior has not been exercised on a real-sized operational area (e.g., a multi-kilometer bay or coastline).

5. **Scripted, non-adversarial vessel obstacles.** The moving vessels in Scenarios D and E follow fixed sinusoidal and straight-line trajectories (Section 4.1; terminological note, Chapter 2) and do not perceive or react to the ego-vessel. This is sufficient to test genuine sensor-to-action reactivity, but it does not yet constitute a fully adversarial or cooperative multi-agent setting.

6. **Limited classical-baseline scope.** The map-dependent family evaluated is restricted to A*, A*(RP), and ACO. Incremental replanners such as D* or D* Lite, and other metaheuristics such as Genetic Algorithms or Particle Swarm Optimization, were not empirically benchmarked; Chapter 2 argues on structural grounds that their shared dependency on a known occupancy grid would not change the qualitative falsification outcome, but this claim remains analytical rather than experimentally confirmed for those specific algorithms.

6.2 Future Work

Building directly on these limitations, future research will pursue: reward regularization to further reduce actuator wear by penalizing control effort derivatives; online Polynomial sail adaptation via meta-reinforcement learning to compensate for hull degradation and real-world sea states; COLREGs-compliant behavior around other vessels by extending the observation space to encode the velocity vectors of surrounding vessels; empirical validation of the computational scaling analysis of Section 4.10 on

a real, multi-kilometer operational area, alongside a direct comparison against an incremental replanner such as D* Lite; and replacing the scripted patrol vessels of Scenarios D and E with independently learned or adversarially trained vessel policies, to evaluate HR-C under genuinely reactive multi-agent conditions rather than multi-vessel ones. The critical next step remains physical deployment: transferring the trained A*/APF+PPO policies to the F-Boat platform through domain randomization and real-world sensor noise filtering to close the gap that high-fidelity simulation alone cannot bridge. The ocean is inherently unstructured and unmapped; this thesis demonstrates that a robotic sailboat can navigate it safely and efficiently using only what it senses in the present moment.

REFERENCES

- AGYEI, Klinsmann; SARHADI, Pouria; POLANI, Daniel. **Deep Reinforcement Learning in Applied Control: Challenges, Analysis, and Insights**. [S.l.: s.n.], 2025. arXiv: [2507.08196](https://arxiv.org/abs/2507.08196) [eess.SY]. Available from: [<https://arxiv.org/abs/2507.08196>](https://arxiv.org/abs/2507.08196).
- ARAÚJO, A. P. D.; AROCA, R. V.; GONÇALVES, L. M. G. Modelagem e Controle de um Barco Autônomo. In: 2013 Workshop of Graduate and Undergraduate Works on Robotics (WGWR-WUWR). Arequipa, Peru: [s.n.], 2013.
- ARAÚJO, André; GUERRA, Raphael, et al. Towards an Autonomous Sailboat Navigation Control Architecture. In: 2024 Latin American Robotics Symposium (LARS). [S.l.]: IEEE, 2024.
- ARAÚJO, André; PREUX, Philippe, et al. A Comparison of DRL with APF and A* with PI Control for Trajectory Planning with Obstacle Avoidance for Sailboat Robots. In: 2024 Latin American Robotics Symposium (LARS). [S.l.]: IEEE, 2024. P. 1–6. DOI: [10.1109/LARS64411.2024.10786465](https://doi.org/10.1109/LARS64411.2024.10786465).
- ARAÚJO, André P. D. et al. Vision of the Seas: Open Visual Perception Framework for Autonomous Sailing Vessels. In: 2023 30th International Conference on Systems, Signals and Image Processing (IWSSIP). [S.l.]: IEEE, 2023.
- ARAÚJO, André Paulo Dantas et al. General System Architecture and COTS Prototyping of an AIoT-Enabled Sailboat for Autonomous Aquatic Ecosystem Monitoring. **IEEE Internet of Things Journal**, IEEE, v. 11, n. 3, 2024. DOI: [10.1109/JIOT.2023.3324525](https://doi.org/10.1109/JIOT.2023.3324525).
- BINGHAM, Brian et al. Virtual RobotX (VRX): A Maritime Robot Simulation and Competition Platform. **IEEE Journal of Oceanic Engineering**, IEEE, v. 44, n. 1, p. 219–232, 2019.
- CHACÓN MOSQUERA, Esteban Francisco. **Autonomous Sailboat Prototype Sensors and Electronics Implementation with Machine Learning for Navigation**. 2021. MA thesis – Universitat Politècnica de Catalunya.
- CORREA, Wanderson et al. Energy Management in Autonomous Vessels Using Restricted Boltzmann Machine. In: 2021 Latin American Robotics Symposium (LARS), 2021 Brazilian Symposium on Robotics (SBR), and 2021 Workshop on Robotics in Education (WRE). [S.l.]: IEEE, 2021.

- CRUZ, Nuno A; ALVES, José C. Auto-heading controller for an autonomous sailboat. In: IEEE. OCEANS 2010 IEEE-Sydney. [S.l.: s.n.], 2010. P. 1–6.
- DE NEGREIROS, Álvaro P. F. et al. Autonomous Sailing Behavioral Learning high level Control Architecture. In: 2023 Latin American Robotics Symposium (LARS), 2023 Brazilian Symposium on Robotics (SBR), and 2023 Workshop on Robotics in Education (WRE). [S.l.: s.n.], 2023. P. 147–152. DOI: [10.1109/LARS/SBR/WRE59448.2023.10333010](https://doi.org/10.1109/LARS/SBR/WRE59448.2023.10333010).
- DORIGO, Marco; MANIEZZO, Vittorio; COLORNI, Alberto. Ant system: optimization by a colony of cooperating agents. **IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)**, IEEE, v. 26, n. 1, p. 29–41, 1996.
- FARAMA FOUNDATION. **Gymnasium: A Toolkit for Developing and Comparing Reinforcement Learning Algorithms**. [S.l.: s.n.], 2023. Available from: <https://github.com/Farama-Foundation/Gymnasium>.
- FOSSEN, Thor I. **Handbook of marine craft hydrodynamics and motion control**. New Jersey, USA: John Wiley & Sons, 2011.
- HART, Peter E; NILSSON, Nils J; RAPHAEL, Bertram. A formal basis for the heuristic determination of minimum cost paths. **IEEE Transactions on Systems Science and Cybernetics**, IEEE, v. 4, n. 2, p. 100–107, 1968.
- JIANG, Lingling et al. A human-like collision avoidance method for autonomous ship with attention-based deep reinforcement learning. **Ocean Engineering**, Elsevier, v. 264, p. 112378, 2022.
- JORGE, Vitor AM et al. A survey on unmanned surface vehicles for disaster robotics: Main challenges and directions. **Sensors**, MDPI, v. 19, n. 3, p. 702, 2019.
- KHATIB, Oussama. Real-time obstacle avoidance for manipulators and mobile robots. **The International Journal of Robotics Research**, Sage Publications, v. 5, n. 1, p. 90–98, 1986.
- KOENIG, Sven; LIKHACHEV, Maxim. D* Lite. In: PROCEEDINGS of the AAAI Conference on Artificial Intelligence. [S.l.: s.n.], 2002. P. 476–483.
- KRAUTWIG, Björn et al. Compensating Environmental Disturbances in Maritime Path Following Using Deep Reinforcement Learning. **Journal of Marine Science and Engineering**, v. 14, n. 4, p. 327, 2026. DOI: [10.3390/jmse14040327](https://doi.org/10.3390/jmse14040327).
- LIU, Lei et al. High-Level Path Planning of Unmanned Sailboat for Sailing Championship and Innovative Education. In: IEEE. 2022 IEEE International Conference on Unmanned Systems (ICUS). [S.l.: s.n.], 2022. P. 1557–1562.

- LUO, Ziyuan; NGUYEN, Minh; YAN, Weiqi. Sailboat Detection Based on Automated Search Attention Mechanism and Deep Learning Models. In: IEEE. 2021 36th International Conference on Image and Vision Computing New Zealand (IVCNZ). [S.l.: s.n.], 2021. P. 1–6.
- MANKINA, Vishali et al. AI-Based Autonomous Sailboat Navigation: A Review. **Journal of Field Robotics**, Wiley, v. 42, p. 4157–4180, 2025. DOI: [10.1002/rob.70004](https://doi.org/10.1002/rob.70004).
- MENESES SANTOS, Lidia G. D. et al. Generative-Augmented Maritime Navigation Dataset: A Deep Synthetic Data Generation Framework. In: 2026 Brazilian Conference on Robotics (CROS). [S.l.: s.n.], 2026. P. 61–66. DOI: [10.1109/CROS69211.2026.11565537](https://doi.org/10.1109/CROS69211.2026.11565537).
- MÉXAS, Rodrigo Picinini; LETA, Fabiana Rodrigues; CLUA, Esteban Walter Gonzalez. Comparison of Reinforcement and Imitation Learning algorithms in autonomous sailboat Digital Twins. **IEEE Latin America Transactions**, IEEE, v. 20, n. 9, p. 2153–2161, 2022.
- MÜLLER, Marlon et al. **Falsification-Driven Reinforcement Learning for Maritime Motion Planning**. [S.l.: s.n.], 2025. DOI: [10.48550/arXiv.2510.06970](https://doi.org/10.48550/arXiv.2510.06970). arXiv: [2510.06970](https://arxiv.org/abs/2510.06970) [eess.SY].
- NEGREIROS, Alvaro P. F. et al. Sustainable Solutions for Sea Monitoring With Robotic Sailboats: N-Boat and F-Boat Twins. **Frontiers in Robotics and AI**, Frontiers Media, v. 9, 2022. DOI: [10.3389/frobt.2022.788212](https://doi.org/10.3389/frobt.2022.788212).
- PARAVISI, Marcelo et al. Unmanned surface vehicle simulator with realistic environmental disturbances. **Sensors**, v. 19, n. 5, p. 1068, 2019.
- RAFFIN, Antonin et al. Stable-baselines3: Reliable reinforcement learning implementations. **The Journal of Machine Learning Research**, v. 22, n. 1, p. 12348–12355, 2021.
- RODRIGUEZ, Renato. **Optimal Control of the AC75 Sailboat for the America’s Cup Race**. 2021. MA thesis – Temple University. Available from: <http://dx.doi.org/10.34944/dspace/7177>.
- RUAN, Xiaobo; DAI, Jiao; YU, Li. Autonomous robot navigation based on a hierarchical cognitive model. **Robotica**, Cambridge University Press, v. 41, n. 5, p. 587–603, 2023.
- S. JÚNIOR, A. G. et al. N-BOAT: An Autonomous Robotic Sailboat. In: 2013 Latin American Robotics Symposium and Competition. Arequipa, Peru: IEEE, 2013. P. 24–29. DOI: [10.1109/LARS.2013.52](https://doi.org/10.1109/LARS.2013.52).
- SALGADO, Alex; BEZERRA, Arthur Andrade, et al. On the Use of Visual Perception for Unmanned Surface Vehicles Navigation. In: 2024 Latin American Robotics Symposium (LARS). [S.l.: s.n.], 2024. P. 1–6. DOI: [10.1109/LARS64411.2024.10786469](https://doi.org/10.1109/LARS64411.2024.10786469).

- SALGADO, Alex; VASCONCELLOS, Eduardo Charles, et al. USV-3.0: Cognitive maritime navigation through vision-language models, Human-in-the-Loop learning, and spatio-temporal memory. **Ocean Engineering**, v. 355, p. 125010, 2026. ISSN 0029-8018. DOI: <https://doi.org/10.1016/j.oceaneng.2026.125010>. Available from: <https://www.sciencedirect.com/science/article/pii/S0029801826008449>.
- SANTOS, D. et al. Design and Implementation of a Control System for a Sailboat Robot. **Robotics**, MDPI, v. 5, n. 1, p. 5, 2016. DOI: [10.3390/robotics5010005](https://doi.org/10.3390/robotics5010005).
- SELLERS, T. et al. Dynamically-updating hierarchical framework for autonomous vehicles in unstructured and dynamic environments. **Sensors and Artificial Intelligence**, v. 2, n. 1, p. 1–16, 2026.
- SMIRNOV, Nikita; TOMFORDE, Sven. Navigation Support for an Autonomous Ferry Using Deep Reinforcement Learning in Simulated Maritime Environments. In: 2022 IEEE Conference on Cognitive and Computational Aspects of Situation Management (CogSIMA). [S.l.: s.n.], 2022. P. 142–149.
- SMITH, R. et al. Reinforcement learning-based obstacle avoidance strategy with a coarse-to-fine approach for autonomous sailboats. **Robotics and Autonomous Systems**, Elsevier, v. 150, p. 104000, 2022.
- STENTZ, Anthony. Optimal and efficient path planning for partially-known environments. In: IEEE. PROCEEDINGS of the 1994 IEEE International Conference on Robotics and Automation. [S.l.: s.n.], 1994. P. 3310–3317.
- SUTTON, Richard S.; BARTO, Andrew G. **Reinforcement Learning: An Introduction**. 2nd. [S.l.]: MIT Press, 2018.
- THE MATHWORKS, INC. **MATLAB and Simulink Release 2024a**. Natick, Massachusetts, 2024.
- TINOCO, Vítor; MALHEIRO, Benedita; SILVA, Manuel F. Design, Modeling, and Simulation of a Wing Sail Land Yacht. **Applied Sciences**, MDPI, v. 11, n. 6, 2021.
- VASCONCELLOS, Eduardo Charles; NEGREIROS, Álvaro Pinto Fernandes, et al. Yara: An Ocean Virtual Environment for Research and Development of Autonomous Sailing Robots and Other Unmanned Surface Vessels. **Journal of Intelligent & Robotic Systems**, v. 111, n. 3, p. 78, 2025. DOI: [10.1007/s10846-024-02212-1](https://doi.org/10.1007/s10846-024-02212-1). Available from: <https://doi.org/10.1007/s10846-024-02212-1>.

- VASCONCELLOS, Eduardo Charles; SAMPAIO, Ronald M., et al. Reinforcement-learning Robotic Sailboats: Simulator and Preliminary Results. In: NEURIPS 2023 Workshop on Robot Learning: Pretraining, Fine-Tuning, and Generalization with Large Scale Models. New Orleans, United States: [s.n.], 2023.
- WANG, L. et al. Improved velocity obstacle method for unmanned surface vehicles real-time trajectory planning. **Applied Ocean Research**, Elsevier, v. 120, p. 103000, 2022.
- WANG, Sijia; JIANG, Hao; WANG, Zhaoqi. Resilient Navigation Among Dynamic Agents with Hierarchical Reinforcement Learning. **SpringerLink**, Springer, 2023.
- WOOLDRIDGE, Michael. **An Introduction to MultiAgent Systems**. 2nd. Chichester, UK: John Wiley & Sons, 2009.
- YANG, Y.; SHEN, W. Millimeter-wave radar-based obstacle detection and tracking method for unmanned sailboats. **IEEE Sensors Journal**, IEEE, v. 22, n. 4, p. 3560–3570, 2022.
- ZHANG, H. et al. A control strategy for autonomous sailboats integrating sensor fusion and adaptive control techniques. **Journal of Marine Science and Technology**, v. 27, p. 1–15, 2022.
- ZHANG, H. et al. Advancing ship automatic navigation strategy with prior knowledge and hierarchical penalty in irregular obstacles: a reinforcement learning approach to enhanced efficiency and safety. **Frontiers in Marine Science**, v. 12, p. 1598380, 2025. DOI: [10.3389/fmars.2025.1598380](https://doi.org/10.3389/fmars.2025.1598380).
- ZHANG, J. et al. Path planning optimization for unmanned sailboat in complex marine environment. **Ocean Engineering**, Elsevier, v. 269, p. 113475, 2022.
- ZHENG, F. et al. A Dynamic Obstacle Avoidance Method for Unmanned Surface Vehicle under the International Regulations for Preventing Collisions at Sea. **Journal of Marine Science and Engineering**, MDPI, v. 10, n. 7, p. 901, 2022.
- ZHOU, Chang et al. Obstacle avoidance strategy for autonomous vessels using a numerical simulation framework. **Ocean Engineering**, Elsevier, v. 234, p. 109288, 2021.