Sim2Sea: Sim-to-Real Policy Transfer for Maritime Vessel Navigation in Congested Waters

Xinyu Cui
Institute of Automation, CAS
School of Artificial Intelligence, UCAS
Beijing, China
cuixinyu2021@ia.ac.cn

Xuanfa Jin
Institute of Automation, CAS
School of Artificial Intelligence, UCAS
Beijing, China
jinxuanfa2022@ia.ac.cn

Xue Yan
Institute of Automation, CAS
School of Artificial Intelligence, UCAS
Beijing, China
yanxue2021@ia.ac.cn

Yongcheng Zeng
Institute of Automation, CAS
Beijing, China
zengyongcheng2022@ia.ac.cn

Luoyang Sun
Institute of Automation, CAS
Beijing, China
sunluoyang2022@ia.ac.cn

Siying Wei
Institute of Automation, CAS
Beijing, China
weisiying2022@ia.ac.cn

Ruizhi Zhang
Institute of Automation, CAS
Beijing, China
ruizhi.zhang@ia.ac.cn

Jian Zhao
Zhongguancun Academy
Beijing, China
jianzhao@zgci.ac.cn

Haifeng Zhang*
Institute of Automation, CAS
Beijing, China
haifeng.zhang@ia.ac.cn

Jun Wang
University College London
London, United Kingdom
jun.wang@cs.ucl.ac.uk

ABSTRACT

Autonomous navigation in congested maritime environments is a critical capability for a wide range of real-world applications. However, it remains an unresolved challenge due to complex vessel interactions and significant environmental uncertainties. Existing methods often fail in practical deployment due to a substantial sim-to-real gap, which stems from imprecise simulation, inadequate situational awareness, and unsafe exploration strategies. To address these, we propose **Sim2Sea**, a comprehensive framework designed to bridge simulation and real-world execution. Sim2Sea advances in three key aspects. First, we develop a GPU-accelerated parallel simulator for scalable and accurate maritime scenario simulation. Second, we design a dual-stream spatiotemporal policy that handles complex dynamics and multi-modal perception, augmented with a velocity-obstacle-guided action masking mechanism to ensure safe and efficient exploration. Finally, a targeted domain randomization scheme helps bridge the sim-to-real gap. Simulation results demonstrate that our method achieves faster convergence and safer trajectories than established baselines. In addition, our policy trained purely in simulation successfully transfers zero-shot to a 17-ton unmanned vessel operating in real-world congested waters. These results validate the effectiveness of Sim2Sea in achieving reliable sim-to-real transfer for practical autonomous maritime navigation.

*Corresponding to Haifeng Zhang <haifeng.zhang@ia.ac.cn>.

Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), C. Amato, L. Dennis, V. Mascardi, J. Thangarajah (eds.), May 25 – 29, 2026, Paphos, Cyprus. © 2026 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). https://doi.org/10.65109/CMRP6518

KEYWORDS

Sim-to-Real Transfer, Maritime Autonomous Navigation, Reinforcement Learning, Innovative Applications

ACM Reference Format:
Xinyu Cui, Xuanfa Jin, Xue Yan, Yongcheng Zeng, Luoyang Sun, Siying Wei, Ruizhi Zhang, Jian Zhao, Haifeng Zhang, and Jun Wang. 2026. Sim2Sea: Sim-to-Real Policy Transfer for Maritime Vessel Navigation in Congested Waters. In *Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), Paphos, Cyprus, May 25 – 29, 2026*, IFAAMAS, 9 pages. https://doi.org/10.65109/CMRP6518

1 INTRODUCTION

As global maritime trade intensifies and nautical activities in nearshore congested waters expand, autonomous navigation is becoming critical for enhancing safety and operational efficiency. These environments, such as ports and coastal fairways, are notoriously difficult to navigate due to dense, heterogeneous traffic, complex geographical constraints, and unpredictable environmental forces. This complexity often leads to scenarios where traditional navigation methods fall short. While rule-based systems grounded in the Convention on the International Regulations for Preventing Collisions at Sea (COLREGs) or collision avoidance algorithms like Velocity Obstacles (VO) provide a crucial safety baseline [26, 28], there are still inherent limitations. The ambiguity of rule-based methods in multi-vessel encounters and their inadequacy in handling mixed obstacle types (e.g., ships and static infrastructure) can lead to indecisive or overly conservative actions [20, 43]. This motivates the development of adaptive, learning-based methods capable of flexible decision-making amid variable traffic dynamics.

Reinforcement learning (RL) [15, 21] has emerged as a promising approach for autonomous nautical operations [8, 41]. However, the path to deploying RL in the real world is fraught with significant obstacles. The foremost challenge is the lack of suitable training environments. To our knowledge, there is no open-source, high-performance simulator that can accurately model the complex dynamics and diverse interaction scenarios required for training a capable maritime agent.

Furthermore, even with a hypothetical perfect simulator, several challenges remain. First, agents should navigate using a multimodal and asynchronous stream of data from sources like Automatic Identification System (AIS) signals, radar signals, and nautical charts, making the creation of a coherent environmental understanding a significant perceptual hurdle. Second, the substantial inertia and underactuated nature of vessels introduce complex temporal dynamics. Purely reactive policies might fail as they cannot account for momentum or external forces like currents. Finally, policies trained in even the most detailed simulations often fail when transferred to a physical vessel due to the sim-to-real gap caused by subtle but critical differences in dynamics, sensor noise, and actuation delays, which is a major challenge to practical deployment.

To address these challenges, we present **Sim2Sea**, an integrated framework designed to enable the development and real-world deployment of maritime autonomous navigation agents. Our approach is built upon three core pillars. First, we develop a high-performance parallel simulator with realistic vessel dynamics and continuous-time safety checks. Second, we designed a dual-stream spatiotemporal policy that leverages a Transformer to understand temporal dynamics and a Bird's-Eye-View (BEV) image to process spatial context. This policy is also augmented by a VO-guided action masking mechanism to ensure safe and efficient exploration. Third, we explicitly bridge the sim-to-real gap through targeted domain randomization, ensuring the learned policy is robust to real-world variability. Our main contributions are as follows:

- We introduce a high-speed parallel maritime simulator designed specifically for large-scale RL training, providing a powerful tool for the research community.
- We propose an agent architecture combining a spatiotemporal policy with active action masking, which is shown to be highly effective for stable and safe learning in complex maritime scenarios.
- Trained with a targeted domain randomization scheme, our learning-based policy achieves successful zero-shot deployment on a 17-ton unmanned vessel in open-sea, congested waters. To the best of our knowledge, this is the first successful trial of its kind on a vessel of this scale.

2 RELATED WORKS

2.1 Maritime Vessel Simulator

The domain of maritime simulation has predominantly focused on Unmanned Underwater Vehicles (UUVs), with notable open-source and high-performance examples like UUVSim [40] and DAVE [38]. However, these simulators focus on underwater scenarios. While proprietary simulators often use established hydrodynamic models like the Maneuvering Modeling Group (MMG) [30, 36] or Abkowitz [1] for maritime vessels, they are typically not open-source and are poorly optimized for large-scale RL training.

Recent simulators like AquaNav [25] prioritize visual fidelity at the cost of parallel processing and accurate vessel dynamics, whereas platforms like MARUS [24] and SMaRCSim [17] lack a specific focus on ship characteristics. To address these deficiencies, we propose a parallel simulator designed for maritime vessels. It supports multiple modeling approaches for maritime vessels, such as the MMG and Nomoto models [10], to effectively facilitate RL training. Furthermore, it features comprehensive modeling of sensors, complex marine environments, and dynamic sea states.

2.2 Autonomous Maritime Navigation

Autonomous maritime navigation has traditionally been dominated by rule-based implementations of COLREGs for Maritime Autonomous Surface Ships, which provide interpretable behavior but struggle in dynamic, congested, or unstructured environments [31, 39]. To address these limitations, recent learning-based methods incorporate domain priors while optimizing decision-making under uncertainty. Adaptive systems that couple classical guidance or control with geometric collision-avoidance, such as fuzzy PID with enhanced velocity obstacles, improve multi-ship encounters by exploiting VO structure [9, 42]. RL also has been used to learn collision avoidance policies that respect COLREGs while scaling to complex interactions [29]. To accelerate training and stabilize policy learning, several works integrate prior knowledge into RL and adjust policy entropy to balance safety and exploration under COLREGs constraints [7, 33, 37].

Despite this progress, most efforts that combine ship control with navigation focus on reward shaping grounded in VO or COLREGs, and systematic exploration near complex coastlines, ports, and restricted waterways remains limited. Existing studies often emphasize path planning rather than closed-loop ship control in such settings [18]. Motivated by these gaps, we aim to explore complex terrains and scenes with a control-oriented approach, and to more deeply leverage VO and related structure to assist RL training and online decision-making.

2.3 Sim-to-Real for Navigation Tasks

Sim2real for navigation has progressed rapidly in autonomous driving, aerial vehicles, and mobile robots through a combination of domain randomization, domain adaptation, system identification, and policy distillation [13]. Recent driving and embodied navigation works couple world-model or BEV-based intermediate representations [23] with student–teacher transfer to bridge noisy sensing and actuation delays, while emphasizing simulation speed and robustness rather than only high visual fidelity to improve transfer reliability [32]. Benchmarks and hybrid frameworks [2, 5, 12] demonstrate that policies trained entirely in simulation can zero-shot or few-shot transfer to real robots when actuation dynamics, sensing noise, and goal specification are modeled consistently, with emerging results across navigation scenarios [6].

For surface vessels, sim2real studies are comparatively sparse, and most reported successes occur in constrained or lab-scale environments rather than open sea. One study applies deep RL with sim2real ideas to static obstacle avoidance by increasing simulation

stochasticity but does not perform real-ship deployment [11]. Another line of work demonstrates sim-to-real transfer for small-scale platforms, such as achieving higher speed and energy efficiency for a small surface vessel [3], mapless navigation for a spherical micro-robot in a tank [34], or control for a wind-powered boat in a basin [4].

Current domain challenges include the lack of high-throughput simulators that can accurately model hydrodynamics, and the difficulty of bridging the sim-to-real gap for underactuated vessels. Our work addresses this by first building a high-performance parallel simulator to capture these complex dynamics. Subsequently, we utilize a spatiotemporal network and targeted domain randomization to achieve successful sim-to-real transfer for open-sea navigation on a full-sized vessel.

3 PRELIMINARIES

Problem Formulation. Our work focuses on the problem of autonomous vessel navigation in congested near-shore waters. The primary objective is to develop an agent capable of guiding the vessel to a designated destination while ensuring collision-free operation with respect to both static geographical features and dynamic obstacles. Therefore, to facilitate agent training and policy optimization, we formulate this navigation task as a Partially Observable Markov Decision Process (POMDP) [19], defined by the tuple $\langle \mathcal{S}, \mathcal{A}, \Omega, P, O, R \rangle$, where $\mathcal{S}$, $\mathcal{A}$, and Ω represent the state space, action space, and observation space, respectively. $P : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is the transition function, indicating the transition probability of taking actions at each state. $O : \mathcal{S} \to \Delta(\Omega)$ is the observation function, which describes what information the agent can obtain at each state. And $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, providing an immediate reward r_t at each step t given a state-action pair. Specifically, in our task settings, the state $s \in \mathcal{S}$ includes the vessel's details along with the positions and velocities of all surrounding obstacles. The observations $o \in \mathcal{O}$ are synthesized from simulated nautical charts, Global Navigation Satellite System (GNSS), AIS, and radar, providing the agent with partial information about the environment. And the action space $\mathcal{A}$ is discrete, consisting of eighteen commands for heading changes distributed uniformly over the interval $[-\pi, \pi]$.

At each step, the agent selects a desired heading, and the simulator propels the vessel toward this heading. An episode concludes when the agent successfully reaches the goal, a collision with an obstacle is registered by continuous-time collision detection, or the maximum number of steps is exceeded. The reward function is designed to foster time-efficient and safe navigation by rewarding progress toward the goal while imposing significant penalties for collisions and boundary violations.

Proximal Policy Optimization (PPO). To solve the described POMDP, we utilize PPO, an actor-critic reinforcement learning algorithm that is effective for complex control tasks. The primary objective for the policy network is given by:

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t \left[\min \left(\rho_t(\theta)\hat{A}_t, clip(\rho_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t \right) \right]. \quad (1)$$

Here, $\rho_t(\theta)$ represents the probability ratio between the new and old policies, $\hat{A}_t$ is an estimator of the advantage function at time step t, and ϵ is a hyperparameter that dictates the clipping range.

Our implementation of PPO is tailored for the maritime vessel navigation task, incorporating features such as multimodal observations and state-dependent action masking. These specific adaptations and other implementation details will be discussed in the Methodology section.

4 METHODOLOGY

In this section, we present Sim2Sea, our integrated framework for sim-to-real maritime vessel navigation. As illustrated in Figure 1, our framework comprises three core components: (1) a high-performance parallel simulator for large-scale training; (2) a spatiotemporal decision network that fuses historical data with spatial context for robust control; and (3) a domain randomization strategy to ensure seamless transfer to real-world vessels. We begin by introducing the simulator, which serves as the foundation for policy learning, followed by a detailed description of the decision-making agent's architecture, and finally the sim-to-real transfer techniques.

4.1 Maritime Vessel Navigation Simulator

Inspired by the success of massively parallel simulators in fields like autonomous driving and robotics, we develop a high-throughput simulation environment tailored for maritime vessel navigation. The Sim2Sea's simulator is built on three design principles: high-fidelity modeling, high-performance parallelization, and efficient interaction computation.

Fidelity and Flexibility in Ship Modeling. Sim2Sea distinguishes itself from existing simulators that often rely on simplified kinematics or general-purpose physics engines by providing specialized and physically-grounded vessel dynamics. It supports three distinct motion models to balance fidelity and computational cost. The primary model is a 3 degrees-of-freedom Maneuvering Modeling Group (MMG) dynamics implementation; the MMG equations are given by:

$$\begin{aligned} m(\dot{u} - vr) &= X_H(u, v, r) + X_P(\cdot) + X_R(\cdot), \\ m(\dot{v} + ur) &= Y_H(u, v, r) + Y_P(\cdot) + Y_R(\cdot), \\ I_z\dot{r} &= N_H(u, v, r) + N_P(\cdot) + N_R(\cdot), \end{aligned} \quad (2)$$

where m is the vessel mass and I_z is its moment of inertia about the z-axis. The variables u, v, and r are the surge, sway, and yaw velocities in the body-fixed frame, with $\dot{u}$, $\dot{v}$, and $\dot{r}$ as their respective accelerations. The terms X, Y, and N represent forces and a moment, where the subscripts H, P, and R denote contributions from hydrodynamics, the propeller, and the rudder. For applications demanding less detail, we also provide a Nomoto yaw-response model and a lightweight nonlinear kinematic model. To ensure numerical stability, the simulator supports both semi-implicit Euler and fourth-order Runge-Kutta (RK4) integration, with internal sub-stepping to handle accurate transients.

To align with real-world ship control systems, Sim2Sea not only supports direct maneuvering, but also abstracts low-level commands by incorporating PID controllers. This provides a unified, high-level control interface where the agent can directly command a target speed and heading, simplifying the policy learning task.

High-Performance Parallelization. While specialized simulators focus on detailed maneuvering analysis, they typically lack

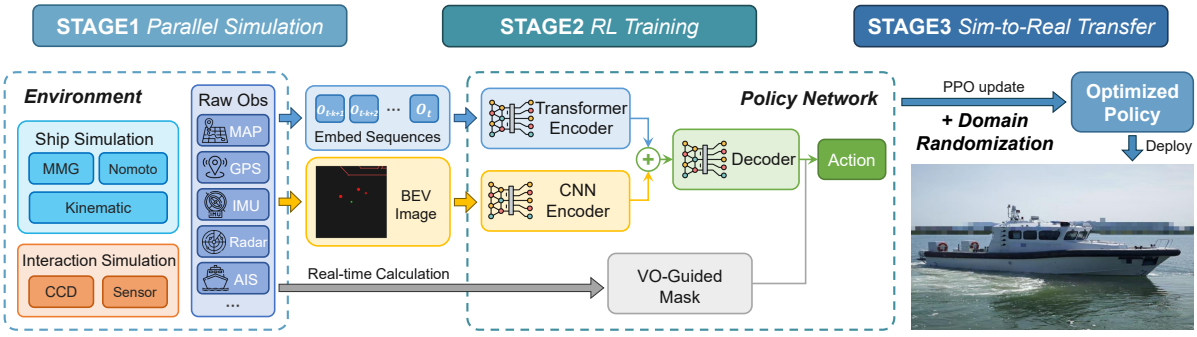


Figure 1: Sim2Sea overview. Left: Parallel maritime simulator with multiple vessel models and interaction simulations. Middle: Spatiotemporal policy with BEV fusion and VO guided active action masking. Right: Zero-shot onboard deployment enabled by domain randomization.

support for parallel execution and RL-centric interfaces. To address this problem, Sim2Sea's simulator is built using the Taichi language [14], enabling native execution on both CPU and GPU backends. We employ an agent-centric parallelization strategy: for a configuration of N environments with M agents each, we execute a single 'struct-for' kernel over the entire set of $N \times M$ agents. This approach avoids serial computation within each environment, ensuring efficient scaling and high throughput, which are crucial for large-scale RL experiments.

Efficient Interaction Computation. Simulating interactions between vessels and with complex geography is computationally intensive. Our environment supports the import of both circular obstacles, representing other vessels from AIS or radar data, and polyline obstacles, representing geographical features like coastlines and breakwaters. Recognizing that interactions are spatially sparse even in congested areas, we use a hash-grid method for broad-phase collision detection to quickly identify potential collisions, significantly reducing unnecessary pairwise checks.

For safety-critical evaluation, we employ continuous-time collision detection (CCD), which checks for intersections along the vessel's entire swept path between discrete steps. This mitigates the risk of missing collisions that can occur with simple discrete-time checks.

4.2 Maritime Decision Network

Our decision network is designed to handle the complexities of maritime environments, such as a variable number of dynamic obstacles and complex coastal geometries. The architecture consists of three key modules: a temporal encoder for understanding vessel dynamics, a spatial encoder for situational awareness, and an innovative action masking mechanism for ensuring safe exploration.

Embedding of the environment and temporal sensing. To address the challenge of variable-sized observations from dynamic obstacles [16], we use an implicit encoding scheme inspired by potential fields. Gradients derived from a logarithmic barrier function are aggregated to form a compact, fixed-size feature vector representing environmental risks. For agent at position $\mathbf{p}$ and an obstacle at distance $d(\mathbf{p})$ with safety radius d_0, the potential energy is defined as $U(d) = \alpha \log(d/d_0)(d_0 - d)^2$, where α is a tunable strength coefficient. The corresponding gradient is:

$$\nabla U(\mathbf{d}) = \alpha \left(2\log(d/d_0) \cdot (d - d_0) + (d - d_0)(1 - d_0/d)\right). \quad (3)$$

We aggregate and normalize gradients from nearby obstacles, computing separate gradients for polyline obstacles, circular obstacles, and a goal-oriented composite term. This encoding compresses high-dimensional, variable-length obstacle data into a compact feature set that captures environmental risk without explicit enumeration. We also integrate the agent state, goal conditions, and other features with the gradients to form the observation o_t.

Since single-step observations are insufficient to capture vessel dynamics under currents and other disturbances, we utilize a transformer encoder to process a sequence of historical observations over the last k time steps, represented as $\mathbf{o}_{t-k+1:t} \in \mathbb{R}^{k \times d}$, where d is the state dimension. This temporal network captures long-range dependencies and recovers underlying environmental dynamics, making the decision more reliable. The temporal feature can be described as: $\mathbf{h}_{\text{temporal}} = \text{Transformer Encoder}(\mathbf{o}_{t-k+1:t})$.

Enhancing the spatial awareness. Simultaneously, to provide the agent with a comprehensive situational spatial picture, we adopt a Bird's Eye View (BEV) representation to encode spatial context around the agent vessel. Unlike terrestrial autonomous vehicles that construct BEV maps through computationally intensive fusion of camera and LiDAR data, maritime platforms benefit from radar and Automatic Identification System (AIS), which directly provide positions of surrounding vessels, and from nautical charts that supply dense geometric information on coastlines and navigational hazards. Leveraging the coordinate information from radar, AIS, and nautical charts, the BEV is constructed through an agent-relative rendering pipeline that incorporates dynamic scaling and depth ordering. This approach enables efficient and robust BEV generation, circumventing the computational complexity and reliability issues of multi-sensor fusion, especially under challenging conditions such as fog or severe vessel motion.

Concurrently, the BEV image is processed into a spatial representation $\mathbf{h}_{\text{spatial}}$ using a lightweight convolutional neural network (CNN) encoder. Finally, temporal and spatial representations are fused and passed through an MLP decoder to generate the actor network output. The decoder produces action logits z over the discrete

action space as

$$z = \text{softmax}\big(\text{Decoder}([\mathbf{h}_{\text{temporal}}; \mathbf{h}_{\text{spatial}}])\big). \quad (4)$$

This dual-stream architecture enables robust decision-making in congested and dynamic maritime environments.

Velocity Obstacles Guided Active Action Mask Generation. While our temporal-spatial network enhances the agent's perception, exploration in geometrically complex environments can still lead to unsafe actions. Unlike implicit guidance methods require extensive trial-and-error, we introduce an active action masking mechanism to explicitly prunes unsafe actions from the policy's output distribution. Our approach dynamically adapts to the environment by using an extended VO method which can handle polyline obstacles by performing intersection checks and distance calculations to identify and mask unsafe headings in real-time.

For each candidate action representing a desired velocity $\mathbf{v}_i$, we perform parallelized safety checks against both types of hazards: For circular obstacles, if $\mathbf{v}_i$ lies within the velocity obstacle cone of any nearby vessel, we calculate the Time-to-Collision (TTC). Any action leading to a TTC below a predefined safety horizon T_h (e.g., $n = 5$ control steps) is deemed unsafe and subsequently masked.

For polyline obstacles, we project the vessel's trajectory over the safety horizon T_h to a future position $\mathbf{p}_{\text{next}}$. First, we calculate the minimum distance between the vessel's predicted trajectory and the line segment to mask an unsafe action. Furthermore, we verify if the vessel's path from its current position $\mathbf{p}$ to $\mathbf{p}_{\text{next}}$ directly intersects with the line segment. If an intersection occurs, the action is also masked. The pseudocode for this mask generation process is detailed in Algorithm 1.

Algorithm 1 Active Action Mask Generation (per timestep)

Input: Current state s, candidate headings $\{a_0 \dots a_{17}\}$, safety horizon T_h
Output: Binary mask m
$m[:] \leftarrow 1$ ▷ Initialize all actions as safe
for i in $0 \dots 17$ **do**
  $v \leftarrow$ heading_to_vel(a_i, ship_speed)
  if $v \in$ VO_cone **then**
    if TTC_with_circular_obstacles(v) $< T_h$ **then**;
      $m[i] \leftarrow 0$; continue
    end if
  end if
  if is_unsafe_wrt_polylines(v, T_h) **then**
    $m[i] \leftarrow 0$; continue
  end if
end for
return m

Concurrently, the final binary mask $\mathbf{m}$ is applied to the logits z_i from the policy network before the softmax operation, producing a safe action distribution for the masked policy π_{masked}:

$$\pi_{\text{masked}}(a_i|\mathbf{s}) = \frac{\exp(z_i) \cdot m_i}{\sum_j \exp(z_j) \cdot m_j}. \quad (5)$$

This mechanism significantly improves sample efficiency, roughly halving the training steps required for convergence while maintaining low collision rates in congested waters.

4.3 Domain Randomization

While the temporal encoding network infers physical dynamics to provide some robustness, a significant gap between simulation and reality persists. To further bridge this sim-to-real gap, we employ domain randomization. This strategy introduces controlled variability into the simulation to force the policy to learn features that are invariant to minor real-world discrepancies.

In Sim2Sea, we apply minor random perturbations to both sensory observations and command transmissions. More critically, we randomize the ocean current model, as it represents a major source of unmodeled dynamics in the real world. This randomization not only enhances the vessel's adaptability to unpredictable environments but also facilitates our temporal network's ability to recover physical information during training.

Based on the observation that near-shore currents often maintain a stable primary direction over short durations, we model the current with two components: a low-frequency dominant flow and a high-frequency random disturbance. At the start of each training episode, the main direction $\mathbf{d}_{\text{main}}$ and peak amplitude A of the current are randomized. The instantaneous velocity of the current is then expressed as:

$$\mathbf{v}_{current} = A \cdot f(t) \cdot \mathbf{d}_{main} + \epsilon \cdot \mathbf{d}_{random}(t). \quad (6)$$

Here, $f(t)$ is a fluctuating multiplier varying in $[0.8, 1.0]$, while the second term represents a minor, time-varying random perturbation $\mathbf{d}_{\text{random}}$ with magnitude ϵ. By exposing the agent to a wide range of such current profiles, we expect it to leverage its temporal network to learn these environmental characteristics and generate appropriate corrective actions, thereby ensuring robust performance upon deployment.

5 EXPERIMENTS

5.1 Parallel Simulation Performance

To evaluate the performance of our simulation architecture, we benchmark it by simulating the 3-DOF MMG vessel dynamics using an RK4 integrator. The benchmark involved executing 50 control commands, each comprising 10 physics sub-steps. We measure the average execution time over 10 trials in a large-scale setup consisting of 1024 environments, each containing 64 agents (a total of 65,536 agents).

We compare three parallelization strategies: Full, Per-Environment, and Per-Agent on both consumer and server-grade hardware. Consumer-grade hardware is characterized by higher clock speeds but fewer compute units, while server-grade hardware typically offers more compute units. As shown in Table 1, our Full Parallelization method is consistently the fastest. The performance gain is most significant on GPUs, where this approach fully utilizes the hardware's parallel processing capabilities. Our design achieves a speedup of over 700-fold on an A100 GPU compared to the slowest baseline (Per-Agent parallelization on CPU), validating its efficiency for large-scale RL training.

5.2 Environmental Setup

We train and evaluate Sim2Sea in two congested water scenarios: Mini Coastline and Mini Port, depicted in Figure 2. The height and width of the maps are both 2000 meters. Each episode initializes

Table 1: Average execution time (in seconds) for simulation with 50 control steps across different parallelization strategies and hardware backends. Lower values indicate better performance. Results are averaged over 10 runs.

Parallelization	Windows PC		Linux Server	
	Ryzen 5900X CPU	RTX 5070Ti GPU	EPYC 7742 CPU	A100 GPU
Full	0.938 ± 0.117	0.036 ± 0.002	0.504 ± 0.011	0.028 ± 0.001
Per-Environment	4.632 ± 0.043	0.147 ± 0.016	9.667 ± 0.037	0.264 ± 0.011
Per-Agent	8.968 ± 0.028	1.399 ± 0.012	20.368 ± 0.194	2.371 ± 0.013

the agent from a random start area and the task is reaching a fixed goal while avoiding obstacles. We instantiate three static circular obstacles at fixed chart locations and three moving circular obstacles whose initial positions, radii, way-point routes, and speeds are randomized per episode. Moving obstacles travel at constant speed along their way-point loops. The agent dynamics follow the simulation model; obstacle motion is purely kinematic and exogenous for simplification.

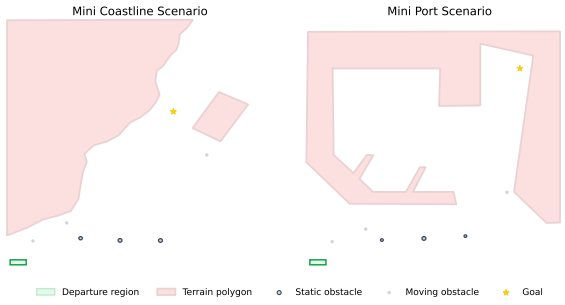


Figure 2: Mini Coastline and Mini Port scenarios. Pink polygons denote land or restricted terrain, green rectangles indicate the departure regions, yellow stars mark the goals, dark circles are static obstacles, and light circles are moving obstacles.

Mini Coastline. This near-shore scenario includes a coastline and rectangular aquaculture no-go zones represented as polylines.

Mini Port. This more challenging scenario requires entering a port through a constrained breakwater opening and reaching an anchorage.

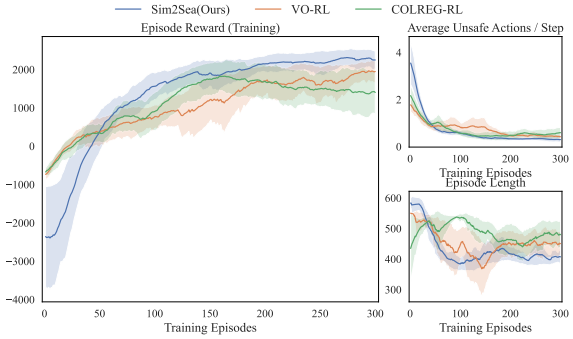
Task protocol. For both scenarios, an episode terminates when the agent reaches the goal region, CCD detects a collision with obstacles, or the step budget is exceeded. The step budget for mini coastline is 600 steps and is 750 steps for mini port. The reward matches the environment implementation: progress shaping proportional to distance-to-goal improvement, a large positive terminal reward upon success, a large negative penalty upon any collision or boundary infringement, a small per-step time cost, and penalties tied to excessive gradient magnitudes as the agent gets too near to obstacles.

5.3 Performance Evaluation in Simulation

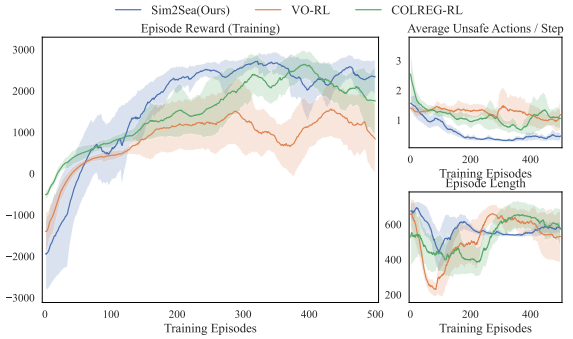
Baseline Comparison. Maritime vessel navigation methods commonly rely on COLREGs or Velocity Obstacles. Prior work often adopts reward shaping to encode safety, so we evaluate **Sim2Sea** against two learning-based baselines that use reward shaping and one classical planner:

- **VO-RL**: integrates VO in reward shaping.
- **COLREG-RL**: integrates COLREGs in reward shaping.
- **VO**: a pure VO controller.

All methods share the same simulator, observation space, discrete action space, and training budget to ensure comparability.



(a) Training curves in the Mini Coastline scenario.



(b) Training curves in the Mini Port scenario.

Figure 3: Learning performance with and without BEV fusion, active action masking and temporal sequence.

For VO-RL, we adapt established shaping rules based on Velocity Obstacles [22, 35]. When there is no collision risk, the reward penalizes the deviation between the selected action and a VO recommended safe heading, which encourages alignment with safe maneuvers. When risk is present, penalties follow standard VO formulations and increase with intrusion into the VO cone and decreasing time to collision.

For COLREG-RL, we apply similar shaping but emphasize regulatory compliance [27]. During collision risk, penalties scale with the distance at closest point of approach (DCPA) and the time to closest point of approach (TCPA), with larger deductions for smaller values that indicate imminent threats. Selecting an action that complies with COLREGs yields a fixed positive reward to reinforce adherence to the rules.

For comparability, we train three seeds per method, select the best checkpoint by validation return, and evaluate ten rollouts per scenario; the VO controller is evaluated ten rollouts. We report success rate, mean episode length, and average unsafe actions encountered per decision step in Table 2.

Sim2Sea uses the same primary reward components as VO-RL and COLREG-RL; minor differences in penalty scaling do not alter

Table 2: Performance of different methods on MiniCoast and MiniPort Scenarios. SR is success rate in percent. Length is mean episode length in steps. UA is the average number of unsafe actions per decision step.

Method	MiniCoast			MiniPort		
	SR(%)	Length	UA	SR(%)	Length	UA
Sim2Sea	93	500 ± 23	1.18 ± 0.42	90	343 ± 25	0.70 ± 0.29
VO-RL	77	558 ± 108	2.24 ± 1.43	47	356 ± 31	1.28 ± 0.37
COLREG-RL	83	587 ± 44	1.96 ± 0.32	77	467 ± 132	0.92 ± 0.34
VO	70	488 ± 22	2.53 ± 0.42	67	314 ± 9	1.24 ± 0.34

qualitative conclusions. As shown in Figure 3 and Table 2, Sim2Sea consistently outperforms the baselines. It achieves the highest success rates and lowest average number of unsafe actions by a significant margin. The superior convergence speed and final performance underscore the advantages of employing VO for active action masking over indirect reward shaping. Moreover, while COLREG-RL is competitive, its rule-based rewards are less adaptive than our direct safety mechanism. The pure VO controller yields short paths but its lower success rate reveals the limitations of purely kinematic models that fail to account for vessel dynamics and actuation delays.

Ablation Studies. To understand the contribution of each component, we trained three variants of our model: one without action masking, one without the BEV input, and one without the temporal sequence encoder. The results in Figure 4 demonstrate that each component is critical for performance.

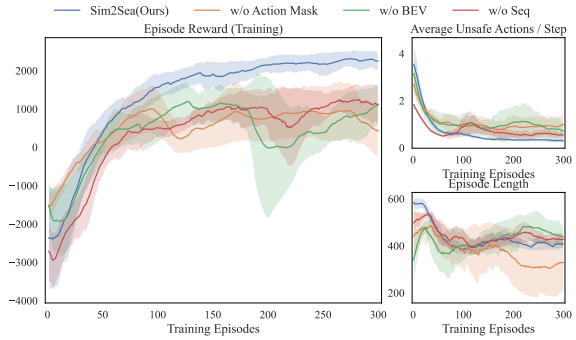
- Without action masking (**w/o Action Mask**): Masking signals are computed but not used to prune actions. The policy samples from the full action set.
- Without BEV input (**w/o BEV**): The BEV branch is removed and the CNN encoder is disabled. The policy uses only the temporal encoder over the vectorized features.
- Without temporal sequence (**w/o Seq**): The temporal encoder is removed. The policy uses the BEV branch and current step features without history.

Each variant is trained on both scenarios with three random seeds and eight parallel environments. We report episode return, average number of masked actions per decision step, and episode length.

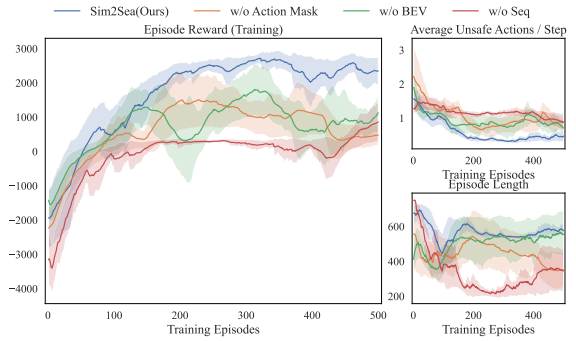
As shown in Figure4, across both scenarios, Sim2Sea consistently achieves higher returns and reaches stable performance in fewer training iterations, which indicates faster convergence. The average unsafe actions per step decreases more rapidly under Sim2Sea, reflecting that the policy remains in safer regions where fewer headings are pruned by VO constraints. This reduction aligns with improved safety because masked actions correspond to headings predicted to violate near term geometric feasibility. Training stability also improves, Sim2Sea exhibits smaller variability across seeds, which suggests that the combination of BEV context and state dependent masking regularizes exploration and produces more predictable learning dynamics.

5.4 Sim-to-Real Deployment

Hardware and Onboard System. Sim2Sea is deployed on a 17-ton unmanned surface vessel powered by twin jet propulsion engines

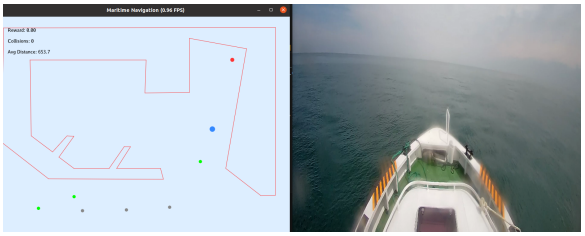


(a) Training curves in the Mini Coastline scenario.

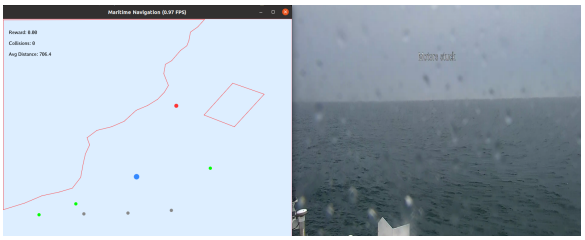


(b) Training curves in the Mini Port scenario.

Figure 4: Learning performance with and without BEV fusion, active action masking and temporal sequence.



(a) Mini Port scenario with forward camera view.



(b) Mini Coastline scenario with side camera view.

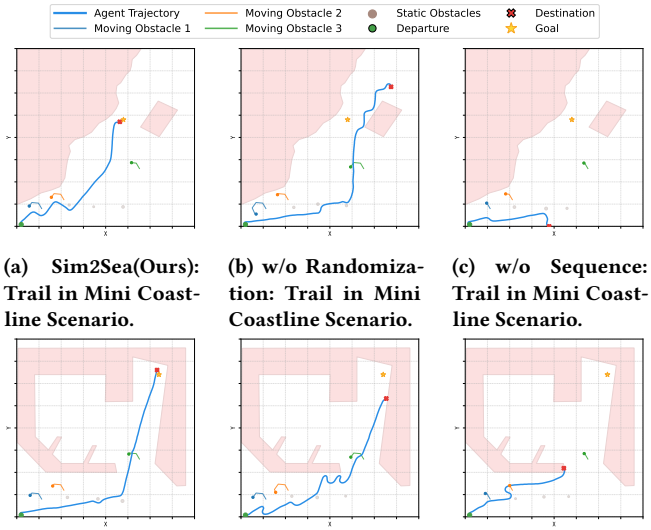
Figure 5: Onboard interface and camera views. Left: graphical user interface displaying vessel state, mission progress, and chart overlays. Right: forward or side camera view providing situational awareness for safety monitoring.

with a maximum speed of 32 knots. The platform supports both manual and autonomous operation, and mode switching can be performed at any time to ensure safety. On board computation is

provided by a compact Linux server with an Intel Core i7 10700 CPU and an NVIDIA RTX 2080 GPU. Due to power and thermal limits, the autonomy stack is designed for modest resource usage and real-time execution.

The perception suite includes Global Navigation Satellite System (GNSS) for positioning, AIS for cooperative vessel tracking, marine radar for noncooperative targets and obstacles, and forward and lateral cameras for situational awareness. These sensors provide inputs to Sim2Sea and support human supervision. The control loop runs at 1 Hz with a fixed speed of 10 knots and controllable headings. A PID-based low-level controller converts heading and speed commands into throttle and rudder signals. Time-stamped logs are recorded for post hoc analysis.

As shown in Figure 5, the onboard computer runs a lightweight graphical interface that visualizes vessel state, mission progress, and chart overlays. Live camera feeds are presented to the safety operator for rapid intervention. A qualified operator remains in the cabin to monitor operations and can switch to manual control whenever risk is detected.



(a) Sim2Sea(Ours): Trail in Mini Coastline Scenario.

(b) w/o Randomization: Trail in Mini Coastline Scenario.

(c) w/o Sequence: Trail in Mini Coastline Scenario.

(d) Sim2Sea(Ours): Trail in Mini Port Scenario.

(e) w/o Randomization: Trail in Mini Port Scenario.

(f) w/o Sequence: Trail in Mini Port Scenario.

Figure 6: Real-ship sim-to-real trials in two scenarios. Post-convergence trajectories compare Sim2Sea with ablations without domain randomization and without temporal inputs. Obstacles are shown in the figures. The departure is green, the destination is marked with a red cross, and the goal is marked with a star.

Zero-Shot Sim-to-Real Transfer Results. We evaluate the zero-shot deployment of Sim2Sea in open-water trials using the optimal policies trained in simulation. Specifically, we selected the best-performing checkpoints after 300 and 500 training episodes in the Mini Coastline and Mini Port scenarios respectively. To unify the observation and control frameworks across domains, we align real-world geodetic data with the simulation's metric space by converting GNSS coordinates into a normalized 2D planar system via Mercator projection. In this planar system, where the positive x-axis represents East and the y-axis North, the vessel's initial position is standardized to the coordinate $[50, 20]$ for all real-world trials. The deployment environment integrates both virtual and real elements: static obstacles and coastlines follow the simulation map layout, while real-world vessels detected via AIS are dynamically incorporated to evaluate safety.

To isolate the effects of domain randomization and temporal modeling, we compare our **Sim2Sea** with two variants: a model trained without domain randomization (**w/o Randomization**), and a model without the temporal encoder that employs an MLP policy (**w/o Sequence**).

A closer examination of the trajectories in Figure 6 reveals the distinct failure modes of the ablated models and underscores the synergistic success of the full Sim2Sea framework. The policy trained without domain randomization (Figures 6b, 6e) exhibits a relatively brittle behavior; its path is characterized by high-frequency oscillations. This indicates that the policy has overfitted to the idealized dynamics of the simulator and is consequently struggling to handle real-world perturbations In contrast, the policy without a temporal encoder (Figures 6c, 6f) fails more catastrophically. Its failure is not one of poor navigation but of fundamental incompetence in controlling the vessel. Lacking a temporal model, the reactive MLP policy is incapable of controlling a system with significant inertia, leading to erratic maneuvers and collisions. The VO mask offers only partial protection, as its linear kinematic assumptions are insufficient for complex, non-linear dynamics.

These distinct failures highlight a crucial insight: successful sim-to-real transfer in this domain requires both **robustness** and **dynamic awareness**, and these two qualities are not independent but deeply intertwined. The temporal encoder allows the agent to learn an internal model of the vessel's dynamics while domain randomization teaches the agent that this internal model is imperfect and that it must constantly adapt to unmodeled external forces. The synergy of these two components creates a policy that is not just reactive, but predictive and adaptive. Sim2Sea does not simply learn a path; it learns to control a complex dynamic system under uncertainty.

6 CONCLUSION

In this paper, we introduce Sim2Sea, a sim-to-real maritime vessel navigation framework that integrates a high-throughput simulator, a dual-stream spatiotemporal policy with BEV fusion and Transformer-based temporal encoding, and VO-guided active action masking, to deliver robust decision-making in congested waters. Experimental results demonstrate that Sim2Sea achieves faster convergence, higher success rates, and fewer unsafe actions compared to baseline approaches.

Notably, Sim2Sea enables successful zero-shot transfer to a real unmanned surface vessel. During deployment, the vessel shows collision-free, smooth, and goal-oriented navigation. Future work will focus on leveraging Sim2Sea's simulation capabilities and its inherent support for multi-environment and multi-agent settings to pursue more challenging tasks in both simulated training and real-world deployment scenarios.

ACKNOWLEDGMENTS

This work was supported by the National Science and Technology Major Project 2022ZD0116404.

REFERENCES

[1] Martin A Abkowitz. 1980. *Measurement of hydrodynamic characteristics from ship maneuvering trials by system identification.* Technical Report.
[2] Rana Azzam, Mohamad Chehadeh, Oussama Abdul Hay, Igor Boiko, and Yahya Zweiri. 2022. Learning to navigate through reinforcement across the sim2real gap. *Authorea Preprints* (2022).
[3] Luis FW Batista, Junghwan Ro, Antoine Richard, Pete Schroepfer, Seth Hutchinson, and Cedric Pradalier. 2024. A deep reinforcement learning framework and methodology for reducing the sim-to-real gap in asv navigation. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 1258–1264.
[4] Kiki Johanna Alberdina Bink, Gabriel David Weymouth, and Bulent DUZ. 2024. Autonomous Sailing with Sim-to-Real Reinforcement Learning. *Available at SSRN 5339032* (2024).
[5] Guillaume Bono, Hervé Poirier, Leonid Antsfeld, Gianluca Monaci, Boris Chidlovskii, and Christian Wolf. 2024. Learning to navigate efficiently and precisely in real environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 17837–17846.
[6] Changan Chen, Jordi Ramos, Anshul Tomar, and Kristen Grauman. 2024. Sim2real transfer for audio-visual navigation with frequency-adaptive acoustic field prediction. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 8595–8602.
[7] Guoquan Chen, Zike Huang, Weijun Wang, and Shenhua Yang. 2024. A novel dynamically adjusted entropy algorithm for collision avoidance in autonomous ships based on deep reinforcement learning. *Journal of Marine Science and Engineering* 12, 9 (2024), 1562.
[8] Yunsheng Fan, Zhe Sun, and Guofeng Wang. 2022. A novel reinforcement learning collision avoidance algorithm for USVs based on maneuvering characteristics and COLREGs. *Sensors* 22, 6 (2022), 2099.
[9] Paolo Fiorini and Zvi Shiller. 1998. Motion planning in dynamic environments using velocity obstacles. *The international journal of robotics research* 17, 7 (1998), 760–772.
[10] VA Golikov, VV Golikov, Ya Volyanskaya, O Mazur, and O Onishchenko. 2018. A simple technique for identifying vessel model parameters. In *IOP Conference Series: Earth and Environmental Science*, Vol. 172. IOP Publishing, 012010.
[11] Changgyu Han, Sekil Park, and Joohyun Woo. 2025. Robust Collision Avoidance for ASVs Using Deep Reinforcement Learning with Sim2Real Methods in Static Obstacle Environments. *Journal of Marine Science and Engineering* 13, 9 (2025), 1727.
[12] Li Haoran, Liu Shasha, Ma Mingjun, Hu Guangzheng, Chen Yaran, and Zhao Dongbin. 2023. NeuronsGym: A Hybrid Framework and Benchmark for Robot Tasks with Sim2Real Policy Learning. *arXiv preprint arXiv:2302.03385* (2023).
[13] Sebastian Höfer, Kostas Bekris, Ankur Handa, Juan Camilo Gamboa, Melissa Mozifian, Florian Golemo, Chris Atkeson, Dieter Fox, Ken Goldberg, John Leonard, et al. 2021. Sim2real in robotics and automation: Applications and challenges. *IEEE transactions on automation science and engineering* 18, 2 (2021), 398–400.
[14] Yuanming Hu, Tzu-Mao Li, Luke Anderson, Jonathan Ragan-Kelley, and Frédo Durand. 2019. Taichi: a language for high-performance computation on spatially sparse data structures. *ACM Transactions on Graphics (TOG)* 38, 6 (2019), 1–16.
[15] Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. 1996. Reinforcement learning: A survey. *Journal of artificial intelligence research* 4 (1996), 237–285.
[16] Ioannis Karamouzas, Nick Sohre, Rahul Narain, and Stephen J Guy. 2017. Implicit crowds: Optimization integrator for robust crowd simulation. *ACM Transactions on Graphics (TOG)* 36, 4 (2017), 1–13.
[17] Mart Kartašev, David Dörner, Özer Özkahraman, Petter Ögren, Ivan Stenius, and John Folkesson. 2025. SMaRCSim: Maritime Robotics Simulation Modules. *arXiv preprint arXiv:2506.07781* (2025).
[18] Jisoo Kim, Byeonggong Hwang, Gi-Hyun Kim, and Ung-Gyu Kim. 2024. Advancing maritime route optimization: using reinforcement learning for ensuring safety and fuel efficiency. *International Journal of e-Navigation and Maritime Economy* 23 (2024), 413–51.
[19] Vikram Krishnamurthy. 2016. *Partially observed Markov decision processes.* Cambridge university press.
[20] Yoshiaki Kuwata, Michael T Wolf, Dimitri Zarzhitsky, and Terrance L Huntsberger. 2013. Safe maritime autonomous navigation with COLREGS, using velocity obstacles. *IEEE Journal of Oceanic Engineering* 39, 1 (2013), 110–119.
[21] Yuxi Li. 2017. Deep reinforcement learning: An overview. *arXiv preprint arXiv:1701.07274* (2017).
[22] Yuqin Li, Defeng Wu, Hongdong Wang, and Jiankun Lou. 2025. Dynamic collision avoidance for maritime autonomous surface ships based on deep Q-network with velocity obstacle method. *Ocean Engineering* 320 (2025), 120335.
[23] Chen Liu, Kiran Lekkala, and Laurent Itti. 2023. World model based sim2real transfer for visual navigation. In *NeurIPS Robot Learning Workshop*, Vol. 2.
[24] Ivan Lončar, Juraj Obradović, Natko Kraševac, Luka Mandić, Igor Kvasić, Fausto Ferreira, Vladimir Slošić, Đula Nađ, and Nikola Mišković. 2022. Marus-a marine robotics simulator. In *OCEANS 2022, Hampton Roads*. IEEE, 1–7.
[25] Pruthviraj Mane, Allen George, Aditya Khonde, and Suresh Sundaram. 2025. AquaNav-Unmanned Surface Vehicle Simulator Using Unreal Engine 5. In *OCEANS 2025 Brest*. IEEE, 1–5.
[26] Jesús A García Maza and Reyes Poo Argüelles. 2022. COLREGs and their application in collision avoidance algorithms: A critical analysis. *Ocean Engineering* 261 (2022), 112029.
[27] Eivind Meyer, Amalie Heiberg, Adil Rasheed, and Omer San. 2020. COLREG-compliant collision avoidance for unmanned surface vehicle using deep reinforcement learning. *Ieee Access* 8 (2020), 165344–165364.
[28] Wasif Naeem, George W Irwin, and Aolei Yang. 2012. COLREGs-based collision avoidance strategies for unmanned surface vehicles. *Mechatronics* 22, 6 (2012), 669–678.
[29] Rongjun Pan, Wei Zhang, Shijie Wang, and Shuhua Kang. 2025. Deep reinforcement learning model for Multi-Ship collision avoidance decision making design implementation and performance analysis. *Scientific Reports* 15, 1 (2025), 21250.
[30] Niklas Paulig. 2024. MMG standard model for ship maneuvering. https://github.com/nikpau/mmgdynamics.
[31] Thomas Statheros, Gareth Howells, and Klaus McDonald Maier. 2008. Autonomous ship collision avoidance navigation concepts, technologies and techniques. *The journal of Navigation* 61, 1 (2008), 129–142.
[32] Joanne Truong, Max Rudolph, Naoki Harrison Yokoyama, Sonia Chernova, Dhruv Batra, and Akshara Rai. 2023. Rethinking sim2real: Lower fidelity simulation leads to higher sim2real transfer in navigation. In *conference on Robot Learning*. PMLR, 859–870.
[33] Chengbo Wang, Xinyu Zhang, Zaili Yang, Musa Bashir, and Kwangil Lee. 2023. Collision avoidance for autonomous ship using deep reinforcement learning and prior-knowledge-based approximate representation. *Frontiers in Marine Science* 9 (2023), 1084763.
[34] Ning Wang, Yabiao Wang, Yuming Zhao, Yong Wang, and Zhigang Li. 2022. Sim-to-real: Mapless navigation for USVs using deep reinforcement learning. *Journal of Marine Science and Engineering* 10, 7 (2022), 895.
[35] Zhanteng Xie and Philip Dames. 2023. Drl-vo: Learning to navigate through crowded dynamic scenes using velocity obstacles. *IEEE Transactions on Robotics* 39, 4 (2023), 2700–2719.
[36] Hironori Yasukawa and Yasuo Yoshimura. 2015. Introduction of MMG standard method for ship maneuvering predictions. *Journal of marine science and technology* 20, 1 (2015), 37–52.
[37] Hao Zhang, Jiawen Li, Liang Cao, Shucan Wang, and Ronghui Li. 2025. 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* 12 (2025), 1598380.
[38] Mabel M Zhang, Woen-Sug Choi, Jessica Herman, Duane Davis, Carson Vogt, Michael McCarrin, Yadunund Vijay, Dharini Dutia, William Lew, Steven Peters, et al. 2022. Dave aquatic virtual environment: Toward a general underwater robotics simulator. In *2022 IEEE/OES Autonomous Underwater Vehicles Symposium (AUV)*. IEEE, 1–8.
[39] Xinyu Zhang, Chengbo Wang, Lingling Jiang, Lanxuan An, and Rui Yang. 2021. Collision-avoidance navigation systems for Maritime Autonomous Surface Ships: A state of the art survey. *Ocean Engineering* 235 (2021), 109380.
[40] Zekai Zhang, Jingzehua Xu, Jun Du, Weishi Mi, Ziyuan Wang, Zonglin Li, and Yong Ren. 2024. UUVSim: Intelligent modular simulation platform for unmanned underwater vehicle learning. In *2024 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 1–8.
[41] Luman Zhao and Myung-Il Roh. 2019. COLREGs-compliant multiship collision avoidance based on deep reinforcement learning. *Ocean Engineering* 191 (2019), 106436.
[42] Xingya Zhao, Yixiong He, Liwen Huang, Junmin Mou, Jie Wen, and Ke Zhang. 2025. Adaptive collision avoidance decision system for autonomous ship navigation. *Journal of Marine Engineering & Technology* 24, 2 (2025), 132–146.
[43] Xiang-Yu Zhou, Jin-Jing Huang, Feng-Wu Wang, Zhao-Lin Wu, and Zheng-Jiang Liu. 2020. A study of the application barriers to the use of autonomous ships posed by the good seamanship requirement of COLREGs. *The Journal of Navigation* 73, 3 (2020), 710–725.