Preprint dated 15 April, 2026.

Online Navigation Planning for Long-term Autonomous Operation of Underwater Gliders

Victor-Alexandru Darvariu[1], Charlotte Z. Reed[1], Jan Stratmann[2], Bruno Lacerda[3], Benjamin Allsup[4], Stephen Woodward[2], Elizabeth Siddle[4], Trishna Saeharaseelan[4], Owain Jones[4], Dan Jones[4], Tobias Ferreira[4], Chloe Baker[4], Kevin Chaplin[4], James Kirk[4], Ashley Iceton-Morris[4], Ryan D. Patmore[4], Jeff Polton[4], Charlotte Williams[4], Christopher D. J. Auckland[5], Rob A. Hall[6], Alexandra Kokkinaki[4], Alvaro Lorenzo Lopez[4], Justin J. H. Buck[4], and Nick Hawes[1]

[1]Oxford Robotics Institute, Department of Engineering Science, University of Oxford, Oxford, UK
[2]Optiver, Amsterdam, The Netherlands
[3]Stateful Robotics, Oxford, UK
[4]National Oceanography Centre, Southampton, UK
[5]School of Environmental Sciences, University of East Anglia, Norwich, UK
[6]Scottish Association for Marine Science, Oban, UK

Corresponding author: Victor-Alexandru Darvariu (email: victord@robots.ox.ac.uk).

The work of Jan Stratmann and Bruno Lacerda was performed while affiliated with the Oxford Robotics Institute. Rob A. Hall was with the School of Environmental Sciences, University of East Anglia for this work.

ABSTRACT Underwater glider robots have become indispensable for ocean sampling, yet fully autonomous long-term operation remains rare in practice. Although stakeholders are calling for tools to manage increasingly large fleets of gliders, existing methods have seen limited adoption due to their inability to account for environmental uncertainty and operational constraints. In this work, we demonstrate that uncertainty-aware online navigation planning can be deployed in real-world glider missions at scale. We formulate the problem as a stochastic shortest-path Markov Decision Process and propose a sample-based online planner based on Monte Carlo Tree Search. Samples are generated by a physics-informed simulator calibrated on real-world glider data that captures uncertain execution of controls and ocean current forecasts while remaining computationally tractable. Our methodology is integrated into an autonomous system for Slocum gliders that performs closed-loop replanning at each surfacing. The system was validated in two North Sea deployments totalling approximately 3 months and 1000 km, representing the longest fully autonomous glider campaigns in the literature to date. Results demonstrate improvements of up to 9.88% in dive duration and 16.51% in path length compared to standard straight-to-goal navigation, including a statistically significant path length reduction of 9.55% in a field deployment.

INDEX TERMS ocean gliders, navigation planning, Monte Carlo Tree Search, long-term autonomy

I. INTRODUCTION

OCEAN sampling is the process of measuring indicators such as temperature, salinity, oxygen, and biomass. It is a crucial task with many applications in the sciences including biology, ecology, oceanography, climatology, and meteorology. Findings based on these observations can inform policy around societal issues such as the management of marine resources and ecosystems. Ocean sampling is inherently challenging, since it involves measuring a dynamic fluid over a range of temporal and spatial scales [1].

Underwater gliders, a type of autonomous underwater vehicle (AUV), are central to addressing the ocean observation problem. In contrast to *static* networks of hydrographic moorings or drifting networks of floats [2], [3], gliders allow for the sampling of *dynamic* phenomena such as eddies or ocean fronts. Gliders are able to operate autonomously for months at a time and can be more energy-efficient and less carbon-intensive than ship operations [4].

Gliders form an important part of the Global Ocean Observation System, and their activity has been increasing in

FIGURE 1: Slocum glider in operation during the eSWEETS[3] field deployment. The proposed *Glider Autonomy Long-term Planning Engine* system was used to pilot gliders autonomously in two long-term deployments totalling approximately 3 months and 1000 km.

the last decade [5]. It is widely accepted that their operation and management need to become more efficient in order for organisations to control increasingly large fleets [6], [7]. Standard operational procedures require input from human pilots upon the surfacing of the glider, which occurs every few hours. Pilots check environmental conditions, the progress of the previous dive, and mission objectives to decide the next control instructions. This raises challenges as piloting supervision is not always available, leading the glider to follow its most recent, possibly outdated, control instructions. Therefore, opportunities for optimising navigation are regularly lost.

A promising direction to optimise navigation autonomously is to leverage navigation planning algorithms, which can plan *online* at every surfacing given an up-to-date surfacing location and environmental conditions. Despite the existence of a fairly large body of work on navigation planning for underwater gliders (reviewed in Section III), until now such methods have not been adopted for autonomous long-term missions. We argue that this is primarily due to the inability of such methods to account for uncertainty in ocean current forecasts and vehicle dynamics.

In this work, we formulate the glider navigation planning problem as a Markov Decision Process (MDP) in which dives correspond to actions, and in which transitions capture uncertainty in the ocean currents and glider motion. Our approach combines a computationally efficient physics-based simulator that generates possible post-dive states with an online sample-based Monte Carlo Tree Search planner. These components are integrated in the *Glider Autonomy Long-term Planning Engine (GALE)* system, which enables online, closed-loop replanning at every surfacing.

GALE was used to conduct two operational deployments in the North Sea towards refining weather forecasts and understanding the environmental impact of offshore wind farms. Our field evaluation, which exceeded 3 months and 1000 km in total, represents the most extensive fully autonomous deployment of online glider navigation planning to date. Results from the field evaluation and simulations demonstrate improved efficiency relative to the standard navigation approach.

Our contributions are as follows:

- **Markov Decision Process formulation for glider navigation planning under uncertainty.** We model multi-dive glider navigation planning as a stochastic shortest-path MDP that explicitly accounts for uncertain ocean current forecasts and glider motion.
- **A computationally efficient, physics-based dive simulator with data-driven calibration.** We design a simulator for MDP state transitions (Figure 4) that is reasonably accurate yet sufficiently fast for sample-based planning. We also propose a principled procedure for setting the simulator parameters based on a dataset of historical real-world glider dives.
- **An online sample-based planning approach for long-term marine autonomy.** We contribute an online sample-based planning technique based on Monte Carlo Tree Search (Figure 5) to solve this MDP and plan over a multi-dive horizon. The planner uses root parallelism and double progressive widening.
- **The design of an autonomous online planning system for Slocum gliders.** We contribute the design of the GALE system, which is underpinned by the proposed methods. It is capable of controlling commercial Slocum gliders [8], [9].
- **Large-scale validation in operational conditions.** We evaluate the approach in two North Sea campaigns totalling over 3 months and 1000 km of autonomous operation, representing the most extensive real-world validation of online autonomous glider navigation planning to date. Compared to standard navigation, in simulation we obtain decreases of up to 9.88% in dive duration and 16.51% in path length on average. We also achieve a statistically significant path length reduction of 9.55% in a field deployment.

II. UNDERWATER GLIDERS

An underwater glider [1] is a type of AUV that moves through the ocean by changing its buoyancy to glide up and down through the water column. Descent and ascent are achieved via a piston or an external bladder, changing the volume of the AUV but keeping mass constant. Movement is performed through *dives* consisting of several *yos*, forming a sawtooth pattern as shown in Figure 2. At the start of a yo, the volume is decreased and the vehicle descends towards the ocean floor; at the bottom of a yo, the volume is increased and the vehicle ascends. The glider's wings convert some of the vertical motion into forwards propulsion. Controlling the angle of the dive is achieved by shifting internal masses

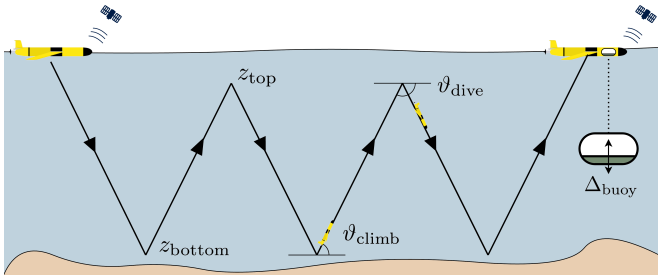

FIGURE 2: An illustration of basic glider operation and the relevant *control parameters* for Slocum gliders. These include the yo floor z_{bottom}, ceiling z_{top}, climb pitch ϑ_{climb}, dive pitch ϑ_{dive}, buoyancy change Δ_{buoy}, and number of yos n_{yos} (3 yos are completed in this dive).

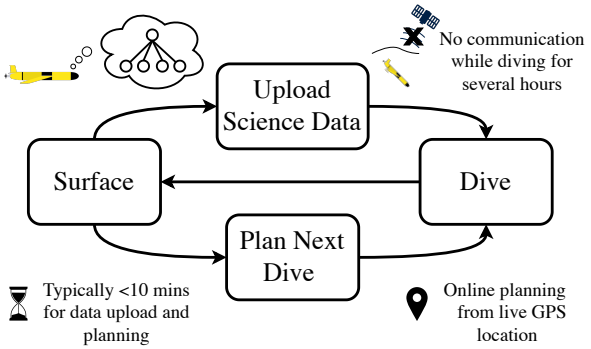

FIGURE 3: The online planning loop of the GALE system and key considerations. *Upload Science Data* and *Plan Next Dive* occur simultaneously while the glider is at the surface.

(typically the battery). Gliders are capable of operating for months at a time given their efficient passive propulsion.

Gliders are equipped with sensors for measuring indicators such as temperature, salinity, and pressure. They also feature a GPS and satellite antenna, allowing them to transmit sampled scientific data and receive *control instructions*. The instructions are comprised of two elements: the *waypoints* that the glider should aim to reach, and the *control parameters* that determine the behaviour of the lower-level controller on the glider during the dive. The control parameters are illustrated in Figure 2.

The online planning loop is illustrated in Figure 3. Control instructions are transferred over satellite while the glider is floating at the surface. The time spent floating is dictated by the amount of scientific data to be transferred and is on the order of 5-10 minutes. This implicitly defines a time budget for determining the next control instructions. An additional challenge is that transfers of control instructions are not fully reliable. Disruptions can occur during periods with adverse weather, and it is common for several transfer failures to happen in a row. The glider will use the latest set of instructions that transferred successfully, which may be significantly out of date.

Standard piloting relies on manually changing the control instructions. In between surfacings, pilots check several data sources indicating glider state and environmental factors such as wind, waves, and ocean currents in order to prepare the next set of control instructions. Given the glider surfaces several times a day for months, it is not feasible for pilots to monitor every surfacing. Despite supervision only being intermittent, estimates from recent deployments suggest that manual piloting accounts for approximately 18% of total campaign costs, which organisations are keen on reducing.

Pilots generally set the closest waypoint such that several dives will be required to achieve it. The glider recomputes the heading towards this waypoint upon every surfacing, a mechanism we refer to as *straight-to-goal (STG)*. By diving straight towards the waypoint, opportunities are regularly lost to optimise considerations such as mission duration (e.g., by setting a heading that exploits favorable currents). This motivates our investigation of online navigation planning techniques, which can adapt to changes in glider state and environmental factors after every surfacing.

III. RELATED WORK

Glider navigation planning has been addressed under several distinct problem formulations. Works based on *optimal control* focus on devising an optimal path under known flow fields. A second category of approaches relies on discrete *Markov Decision Process* formulations that are solved via search or learning. Lastly, another category of works focuses on optimising coordination and coverage for *fleets* of gliders, rather than planning an efficient route to a destination.

A. Optimal Control

Planning a trajectory between two waypoints was studied in [10], where the problem is transformed into a nonlinear programming formulation and solved via numerical optimisation. An extension for handling time-varying currents was introduced in [11]. A path planning approach for estuarine environments (characterised by strong tidal currents) that can optimise for both speed and energy usage was proposed in [12]. Ideas from fluid dynamic equations were combined with control theory in [13] to yield a general method for vehicles traveling through a variable flow field. Most recently, [14] proposed a method for trajectory planning for AUVs with active propulsion using high-resolution ocean forecasts, applying it to an estuarine environment.

A key limitation of such path planning methods is that they assume deterministic motion and environments. As a result, they do not explicitly model uncertainty in ocean current forecasts and the movement of the glider, which can lead to compounding errors over multi-dive horizons. Furthermore, while they can achieve impressive performance in minimising cost to destination, their evaluation is typically limited to simulation. This stands in contrast to our work, which

explicitly models uncertainty and evaluates performance in multi-month field campaigns.

B. Search and Markov Decision Process Formulations

Path planning using a variant of the A* algorithm was proposed in [15]. To make the execution of A* tractable, the method assumes deterministic motion under a single realisation of a 2D current field and imposes a constant dive duration. The work in [16] adopts a Markov Decision Process problem formulation, considers currents across multiple depth levels, and conducts a field trial. However, this approach uses a highly limiting discretisation of the state space, which allows for offline planning using a value iteration algorithm. Uncertainty is incorporated through a Gaussian model over the summed vehicle and current velocities, yielding transition probabilities for each of the 8 surrounding cells. The approach assumes a fixed underlying current field and models uncertainty in a simplified manner, without accounting for forecast uncertainty or state-dependent execution noise. Furthermore, the field experiment is limited to two round-trip traversals between a pair of waypoints separated by 3.5 km.

Very recent work [17] has applied deep reinforcement learning (RL) to glider path planning. The proposed approach uses ocean current forecasts, which are paired with a simplified glider dynamics model to yield a simulation environment. The path planning policy is trained using a Soft Actor-Critic architecture for optimising a complex multi-objective reward. While improvements are demonstrated in simulation over other deep RL variants, important limitations remain: the planning occurs entirely offline and is not adjusted based on the live surfacing location of the vehicle; the method does not incorporate forecast or execution uncertainty; and the field trial is limited to a single trajectory.

C. Fleet Coordination

A control law allowing a formation of three gliders to maintain an equilateral triangle structure was formulated in [18]. It can be used to determine 3-dimensional gradients of ocean properties such as temperature and salinity. A general approach for controlling a mobile sensor network such as a glider fleet was developed in [19]. The latter method was successfully applied in the Adaptive Sampling and Prediction (ASAP) trial [20], which deployed 6 Slocum gliders over 24 days covering a total distance of 3270 km. However, this required 14 interventions from human pilots to adjust the motion patterns based on ocean forecasts. The work [21] considered the problem of planning paths for a fleet of gliders such as to achieve complete coverage of the sampling area. Ant colony optimisation was used for altering preset paths in order to avoid obstacles. In simulation, the method yielded shorter paths with less consumed energy. We note that these approaches focus on the mission-level objectives, and are complementary to our work, which addresses the lower-level problem of navigation planning

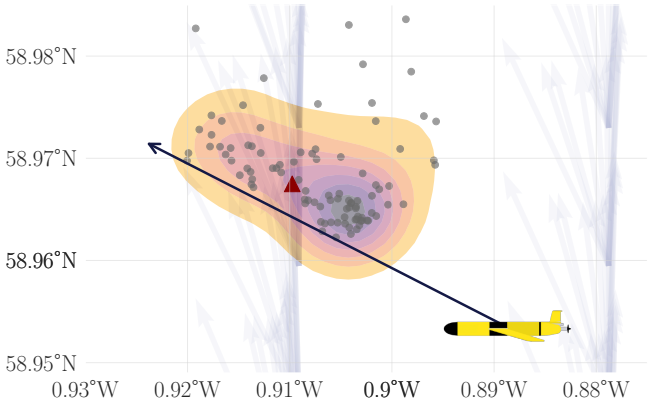

FIGURE 4: An illustration of the key inputs and outputs of the glider dive simulator. The glider icon represents the dive *initial location*, the arrow indicates its *direction*, and background arrows correspond to *ocean current forecasts* over several depth levels. The grey circles are the outputs of the simulator indicating *possible surfacing locations*, i.e., stochastic MDP transitions. The red triangle is the *true surfacing location*.

D. Summary

Overall, our work is the first to couple uncertainty-aware online navigation planning with extended real-world deployments of ocean gliders. To our knowledge, the multi-month campaigns presented here are the most extensive demonstration to date of fully autonomous glider navigation without regular human-in-the-loop interventions.

IV. METHODS

A. Problem Statement

As mentioned, gliders execute dives and surface every few hours. They float while transmitting scientific information and receiving control instructions, after which they dive. During dives, they lack the ability to communicate or localise precisely, rendering the states between dives practically unobservable. Therefore, our model of the problem considers surfacings as discrete events. This enables us to approach the high-level navigation planning problem over a relatively long time horizon, while the control in between dives is handled by a lower-level controller on board the glider.

The glider's position is represented as $\mathbf{p} = (\lambda, \phi)$, where λ and ϕ denote the longitude and latitude respectively. A glider state is denoted as $\mathbf{s} = (\mathbf{p}, \tau)$, where τ is a continuous timestamp in seconds. We refer to states at MDP time t as $\mathbf{s}_t$, and also use $\mathbf{p}_t$ and τ_t as shorthands. We are given a start state $\mathbf{s}_0$ and goal position $\mathbf{p}_{\text{goal}}$ and seek to find a trajectory that minimises the total duration $\Delta\tau_{\text{total}} = \tau_T - \tau_0$ for surfacing within a radius ρ of the goal at the terminal timestep T.

Glider navigation is strongly impacted by *currents*. We represent them as the vector $\mathbf{q}(\lambda, \phi, z, \tau) = [u, v, w]^\top$ denot-

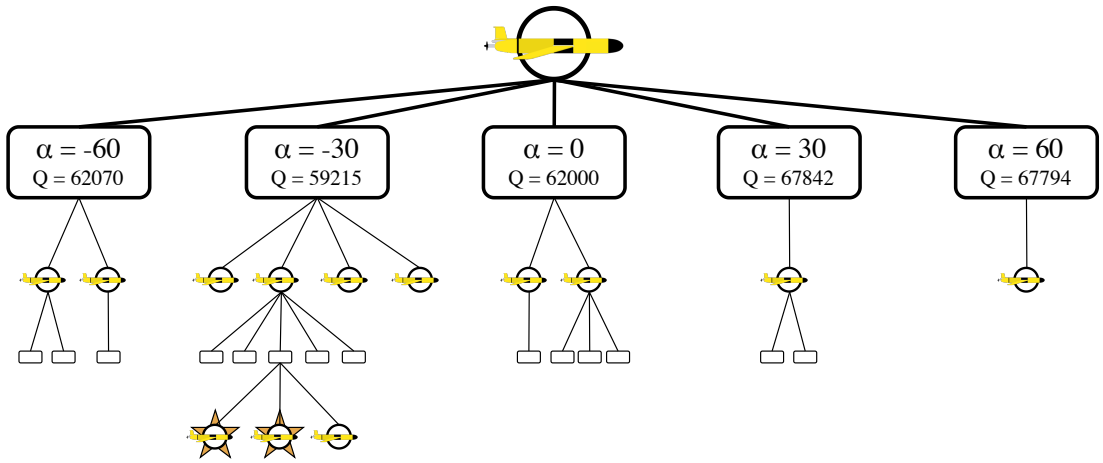

FIGURE 5: An illustration of a search tree built by the planner. Circles represent state nodes and squares represent action nodes. Action nodes can have mulitple children as MDP transitions $\mathcal{T}$ are stochastic. Q is the expected cost of an action. At the root, diving at -30 degrees relative to the goal is estimated to lead to the lowest expected cost. The search tree is several levels deep and explored asymmetrically by the planner to balance exploration and exploitation. Starred nodes denote terminal states in which the glider reaches the goal.

ing the velocity in the eastward, northward, and downward components respectively. Here, z represents the depth. The *bathymetry* $\mathbf{b}(\lambda, \phi) = z_{\text{bathy}}$ identifies the maximum ocean depth at each position and is relevant since gliders typically inflect when approaching the ocean floor. In contrast to currents, which we cannot measure or predict fully accurately, we may assume bathymetry is known and fixed on the timescales of the considered planning problem.

Each instance I of the glider navigation planning problem is therefore defined as the tuple $I = (\mathbf{p}_0, \tau_0, \mathbf{p}_{\text{goal}}, \rho, \mathbf{q}(\cdot), \mathbf{b}(\cdot))$.

B. Markov Decision Process Formulation

MDPs are a widely employed formalisation of decision-making [22], [23]. We define an MDP as the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{C}, \mathcal{G})$, where $\mathcal{S}$ is the *state space* defining all configurations the agent may find itself in, $\mathcal{A}$ is the *action space* defining valid controls, $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ denotes the *transition function* satisfying $\sum_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} \mathcal{T}(s, a, s') = 1 \ \forall s \in \mathcal{S}$, and $\mathcal{C} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, \infty)$ is a *cost function*. Also let $\mathcal{G} \subseteq \mathcal{S}$ denote a set of absorbing *goal states* s.t. $\forall s_g \in \mathcal{G} \wedge s' \notin \mathcal{G}$ we have $\mathcal{T}(s_g, a, s_g) = 1$, $\mathcal{T}(s_g, a, s') = 0$, and $\mathcal{C}(s_g, a, s_g) = 0$. This class of models is known as *stochastic shortest-path (SSP)* MDPs. A deterministic, stationary, Markovian policy is a mapping $\pi : \mathcal{S} \to \mathcal{A}$. The *value function* of such a policy for a state s is defined as $V^\pi(s) = \mathbb{E}[\sum_{t'=0}^{\infty} C^{\pi_s}_{t'+t}]$, where $C^{\pi_s}_{t'+t}$ is a random variable denoting the cost incurred as a result of executing policy π starting in state s for all t. We seek to find the optimal policy π^* which satisfies $V^*(s) = \min_\pi V^\pi(s)$, i.e., the policy that achieves the minimal cost in expectation at each state.

To formulate glider navigation planning as an SSP MDP, we must define each element of $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{C}, \mathcal{G})$. We consider each dive as a discrete decision, hence the MDP time t advances by 1 with each dive.

States. The state space $\mathcal{S}$ is continuous and comprises all positions $\mathbf{p}$ and times τ the glider may be in. In practice, these values are bounded: $\lambda, \phi \in [-180°, 180°]$ while $\tau \geq \tau_0$ and is capped by the maximum planning horizon (typically a few days).

Actions. Actions $\mathcal{A}$ are defined as tuples $\mathbf{a} = (\alpha, \mathbf{c})$, where α denotes the dive direction, while $\mathbf{c}$ is itself a tuple consisting of the dive control parameters. We opt to discretise the dive direction component of the action space. This allows us to incorporate the knowledge that small increments of change in heading are largely irrelevant given the noise in the system, and avoids the challenges of working directly with a continuous action space. Furthermore, we leverage the observation that glider headings ψ are set relative to the bearing β to the goal position $\mathbf{p}_{\text{goal}}$, as illustrated in Figure 6. We note the distinction between the bearing to the goal (a geometric quantity) and the vehicle heading (a control variable). Setting $\psi = \beta$ corresponds to aiming the glider *straight-to-goal* which, as previously mentioned, is the default mechanism by which Slocum gliders set their heading towards a distant waypoint. Instead, we may wish to set a heading that differs from the bearing in order to account for the effect of the currents. Therefore, we allow as actions a discrete set of relative bearings $\alpha = \beta - \psi$. Assuming a north-east-down reference frame, positive values of α correspond to diving clockwise relative to the goal, while negative values correspond to diving anti-clockwise. This preserves a lower branching factor compared to deciding the heading directly,

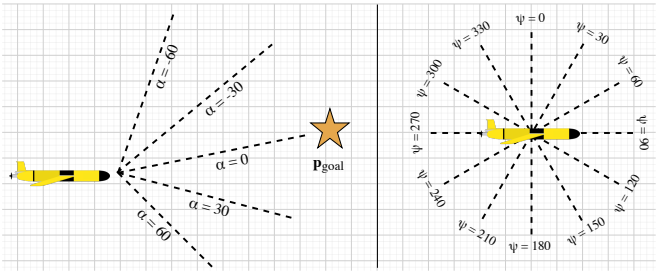

FIGURE 6: Illustration of the proposed action space formulation (left), which uses bearings α that are relative to the goal. This leads to a lower branching factor than deciding headings ψ directly (right) while retaining relevant actions.

while prioritising actions that are more likely to lead to progress towards the goal.

Transitions. Transitions $\mathcal{T}$ govern the probability of the glider surfacing in state $\mathbf{s}_{t+1}$ after diving from state $\mathbf{s}_t$ using action $\mathbf{a}_t$. In this problem, transitions are impacted by uncertain currents $\mathbf{q}$ and glider motion, and cannot be modelled fully accurately. Transitions are therefore stochastic and highly uncertain. To devise navigation plans, we thus require access to a sample-based *simulator* that can generate samples of likely surfacing states. This component is described in depth in Section IV.C.

Costs. The cost function $\mathcal{C}$ we consider is $\mathcal{C}(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1}) = \tau_{t+1} - \tau_t$, i.e., we incur a cost equal to the duration of the dive. Alternative cost definitions may also account for the energy usage of each set of controls; however, in our considered experimental setup, this is approximately constant with respect to time.

Goals. The goals $\mathcal{G}$ are defined as $\mathcal{G} = \{\mathbf{s} \in \mathcal{S} \mid d(\mathbf{p}, \mathbf{p}_{\text{goal}}) \leq \rho\}$, where ρ is a tolerance radius and d is geodesic distance.

Regarding the impact of currents on transitions $\mathcal{T}$, as we cannot accurately measure and predict currents across the entire water column, we must rely on forecasts $\hat{\mathbf{q}}$. Many ocean models exist, which vary in accuracy as well as spatial and temporal resolutions. Such models employ discretisation by splitting coordinates, depth, and time into intervals. Concretely, as we focus our deployment and evaluation in the North Sea region, we make use of the AMM15 [24] model, which provides forecasts at hourly intervals over grid cells of 1.5×1.5 km and 33 levels whose coarseness increases with depth. We obtain these forecasts via the Met Office marine data service [25]. These forecasts have a horizon of 5 days. This model does not make vertical velocities available in the publicly available outputs; therefore, we assume $w = 0$, i.e., $\hat{\mathbf{q}}(\lambda, \phi, z, \tau) = [u, v, 0]^\top$, which is a reasonable assumption in tidally dominated shelf seas.

C. Custom Glider Dive Simulator

Our approach requires a *simulator* that can generate samples from the transition function $\mathcal{T}$. Recall that, as described in Section II, control instructions are decided within a fixed time budget of 5-10 minutes while the glider is floating at the surface. Given that the number of trials that can be run in a fixed time budget strongly influences MCTS performance [26], the simulator must balance the accuracy of sampled transitions with execution speed. Constructing a search tree over a horizon of several dives is crucial for going beyond reactive current correction.

The state-of-the-art open source simulator for Slocum gliders is `glidersim` [27], whose detailed approach takes into account the actuator behaviour and hydrodynamics of Slocum gliders [28], [29]. However, this comes at the expense of costly runtime. The method also does not account for the unknowns surrounding the current forecasts and therefore it is unsuitable for use in our approach. Thus, we set out to devise a custom simulator that is more appropriate for sample-based planning methods.

To do so, we leverage the following observations. Firstly, the majority of large-scale, publicly available ocean models (including AMM15) do not include vertical velocities in the published model outputs. As such, we cannot use forecasts to account for changes in the glider's descent and ascent rate, and we can treat them as approximately constant. While vertical density gradients (stratification) can also influence vertical motion in practice, capturing these effects would require a more detailed hydrodynamic model, which we leave to future work. Secondly, the glider moves in a largely steady state (i.e., constant speed and attitude relative to the water) when performing the sawtooth motion. This is not the case at the bottom and top of each yo, when the onboard controller changes the glider pose and velocity. Nevertheless, these transient phases can be considered negligible in terms of the impact on the overall trajectory, as demonstrated by prior studies [30], [31]. This motivates our simulation approach that chains together steady-state segments.

The pseudocode for our glider dive simulator is given in Algorithm 1. The *inputs* to the simulator are the glider state $\mathbf{s}$, action $\mathbf{a}$, bearing β, current forecasts $\hat{\mathbf{q}}$ and bathymetry $\mathbf{b}$. The *output* of the simulator is the post-dive state $\mathbf{s}'$. We note that, unlike in the MDP states described in Section IV.B, in which the glider is always at the surface, the internal glider states in the simulator also contain a depth level. We denote this internal state as $\mathbf{g}$. Querying the forecast and bathymetry are denoted $\hat{\mathbf{q}}(\mathbf{g})$ and $\mathbf{b}(\mathbf{g})$.

The main loop of the algorithm (`SimulateDive`) simulates the characteristic seesaw-like motion by chaining together upwards and downwards steady-state phases (`SteadyUp` and `SteadyDown`) until the number of yos to perform is exhausted. The `SteadyDown` method simulates a downward steady-state phase by repeatedly propagating the glider to the next forecast cell. The `SteadyUp` method does the same except with the glider moving upwards. The general idea is that, as the forecast currents are representative for a whole grid cell, the glider state can be propagated linearly within that cell until the cell border is hit (`TimeToNextCell`). Glider horizontal speed $s \in \mathbb{R}^+$

Algorithm 1 Custom Glider Dive Simulator

1: **Params:** $\Theta = \{\theta_{\text{drag\_i}}, \theta_{\text{drag\_j}}, \theta_{\text{curr\_mag}}, \theta_{\text{curr\_dir}}, \theta_{\text{curr\_min}}, \theta_{\text{motion\_mag}}, \theta_{\text{motion\_dir}}, \theta_{\text{motion\_min}}\}$
2: **Function** `SimulateDive`$(\mathbf{s}, \mathbf{a}, \beta, \hat{\mathbf{q}}, \mathbf{b})$:
3: $\mathbf{s}' \leftarrow \mathbf{s}$, `halfYosDone` $\leftarrow 0$
4: **while** `halfYosDone` $< 2 * \mathbf{a}.\mathbf{c}.n_{\text{yos}}$ **do**
5: **if** `halfYosDone` mod $2 = 0$ **then**
6: $\mathbf{s}' \leftarrow$ `SteadyDown`$(\mathbf{s}', \mathbf{a}, \beta, \hat{\mathbf{q}}, \mathbf{b})$
7: **else**
8: $\mathbf{s}' \leftarrow$ `SteadyUp`$(\mathbf{s}', \mathbf{a}, \beta, \hat{\mathbf{q}}, \mathbf{b})$
9: `halfYosDone` += 1
10: **return** $\mathbf{s}'$
11: **Function** `SteadyDown`$(\mathbf{s}, \mathbf{a}, \beta, \hat{\mathbf{q}}, \mathbf{b})$:
12: $\psi \leftarrow$ `HeadingFromAction`$(\mathbf{a}.\alpha, \beta)$
13: $s, w_g \leftarrow$ `GliderFlightModel`$(\mathbf{a}.\mathbf{c})$
14: $u_g, v_g \leftarrow s \cdot \sin\psi, s \cdot \cos\psi$
15: $u_g, v_g \leftarrow$ `ApplyMotionNoise`(u_g, v_g)
16: $\mathbf{g} \leftarrow [\mathbf{p}, z, \tau]^\top$
17: $\dot{\mathbf{g}} \leftarrow [u_g, v_g, -w_g]^\top$
18: **while** $\mathbf{g}[2] > \min(\mathbf{a}.\mathbf{c}.z_{\text{bottom}}, \mathbf{b}(\mathbf{g}))$ **do**
19: $\mathbf{g} \leftarrow$ `ToNextCell`$(\hat{\mathbf{q}}, \mathbf{g}, \dot{\mathbf{g}}, \mathbf{a}.\mathbf{c}.z_{\text{bottom}})$
20: $(\mathbf{p}', \tau') \leftarrow (\mathbf{g}[1], \mathbf{g}[3])$
21: **return** $(\mathbf{p}', \tau')$
22: **Function** `ToNextCell`$(\hat{\mathbf{q}}, \mathbf{g}, \dot{\mathbf{g}}, z_{\text{lim}})$
23: $u, v, \_ \leftarrow \hat{\mathbf{q}}(\mathbf{g})$
24: $\tilde{\mathbf{q}}(\mathbf{g}) \leftarrow$ `GetNoisyCurrents`(u, v)
25: $\dot{\mathbf{g}}_{\text{true}} \leftarrow$ `ApplyCurrAccel`$(\dot{\mathbf{g}}, \tilde{\mathbf{q}}(\mathbf{g}), \theta_{\text{drag\_i}}, \theta_{\text{drag\_j}})$
26: $\Delta\tau \leftarrow$ `TimeToNextCell`$(\hat{\mathbf{q}}, \mathbf{g}, \dot{\mathbf{g}}_{\text{true}}, z_{\text{lim}})$
27: $\mathbf{g} \leftarrow \mathbf{g} + \dot{\mathbf{g}}_{\text{true}} \cdot (\Delta\tau + \varepsilon)$
28: **return** $\mathbf{g}$
29: **Function** `SteadyUp`$(\mathbf{s}, \mathbf{a}, \beta, \hat{\mathbf{q}}, \mathbf{b})$:
30: /* Similar to `SteadyDown` but with $+w_g$ in $\dot{\mathbf{g}}$ and condition $\mathbf{g}[2] < \mathbf{a}.\mathbf{c}.z_{\text{top}}$ */
31: **Function** `TimeToNextCell`$(\hat{\mathbf{q}}, \mathbf{g}, \dot{\mathbf{g}}_{\text{true}}, z_{\text{lim}})$
32: /* Calculates time until next forecast cell $\hat{\mathbf{q}}$ is reached using geographical distances and glider speed $\dot{\mathbf{g}}_{\text{true}}$. */
33: **Function** `GetNoisyCurrents`(u, v):
34: $|q| \leftarrow \sqrt{u^2 + v^2}, \alpha_{\text{curr}} \leftarrow \text{atan}(v, u)$
35: `magN` $\sim \mathcal{N}(0, \theta_{\text{curr\_mag}})$, `dirN` $\sim \mathcal{N}(0, \theta_{\text{curr\_dir}})$
36: $|q|$ += `magN`, α_{curr} += `currN`
37: $|q| = \max(|q|, \theta_{\text{curr\_min}})$
38: **return** `polarToVec`$(|q|, \alpha_{\text{curr}})$
39: **Function** `ApplyMotionNoise`(u, v):
40: /* Similar to `GetNoisyCurrents` but with generatingmagN, `dirN` using random walks with parameters $\theta_{\text{motion\_mag}}, \theta_{\text{motion\_dir}}$ and minimum speed $\theta_{\text{motion\_min}}$. */

and vertical speed (rate of change of depth) $w_g \in \mathbb{R}^+$ are calculated (`GliderFlightModel`) using the approach of Merckelbach *et al.* [29]. This takes into account the control parameters in $\mathbf{c}$ as well as water properties. The flight model parameters are fit to historical glider data from the region of interest. Acceleration is applied to the glider speeds as determined by the current forecast and drag coefficients $\theta_{\text{drag\_i}}, \theta_{\text{drag\_j}}$ that are part of the simulator parameters.

By default, transitions $\mathcal{T}$ generated by this simulator would be deterministic. However, as we would like the produced navigation plans to be robust to uncertainty in the forecast and glider motion, we inject *current noise* and *motion noise* in order to yield several plausible samples of transitions given the same inputs.

We use a simple Gaussian noise model for the current noise (`GetNoisyCurrents`). Independent noise is added to the magnitude and direction by sampling from Gaussian distributions with mean 0 and configurable standard deviations $\theta_{\text{curr\_mag}}$, $\theta_{\text{curr\_dir}}$. These perturbations are applied consistently for all forecasts $\hat{\mathbf{q}}$ upon the initialisation of the simulator with a particular random seed, corresponding to a systematic bias in the forecast.

To implement motion noise (`ApplyMotionNoise`), we update the glider velocity with a noise model suggested by glider pilots. Namely, we use a random walk to reflect the noise of the internal dead-reckoning characterised by a very inaccurate state estimation and other local perturbations. We simulate a symmetric random walk over integers with absorbing barriers for which the probability of the value decreasing, staying put, or increasing is uniform. Separate walks are used for the magnitude and direction. The values of the random walk states are multiplied with the $\theta_{\text{motion\_mag}}$ and $\theta_{\text{motion\_dir}}$ noise parameters then added to the glider magnitude and direction.

The simulator parameters Θ, enumerated in Line 1 of Algorithm 1, control the impact of the currents on glider speed and the amount of noise that is injected into the simulator. As these parameters are challenging to establish a priori and depend significantly on the deployment vehicle, location, and conditions, we propose a procedure to tune them using past glider data.

D. Simulator Parameter Tuning

Choosing the simulator parameters Θ appropriately is of crucial importance for ensuring the fidelity of the generated transitions from $\mathcal{T}$ and, consequently, the effectiveness of the generated navigation plans. We denote an instance of the simulator parameterised with a particular Θ as $\mathcal{T}_\Theta$.

To enable tuning Θ, we assume availability of a dataset $\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^N$ of historical dives with entries $\mathbf{x}^{(i)}$ of the form $((\mathbf{s}^{(i)}, \mathbf{a}^{(i)}, \beta^{(i)}, \hat{\mathbf{q}}^{(i)}, \mathbf{b}^{(i)}), \mathbf{s}^{(i)}_{\text{true}})$, comprising the conditions prior to starting a dive and the true post-dive state after the surfacing of the glider. For simplicity, we also assume that all parameters $\mathbf{c}$ except n_{yos} and z_{bottom} are held constant within a dataset used to fit the simulator parameters.

We focus on optimising the generated simulator positions and dive durations relative to the ground truths $\mathbf{p}^{(i)}_{\text{true}}$ and $\Delta\tau^{(i)}_{\text{true}}$ respectively, where $\Delta\tau = \tau' - \tau$. For each $\mathbf{x}^{(i)}$, we draw M samples of future states

$\mathbf{s}^{(i,1)}, \mathbf{s}^{(i,2)}, \ldots, \mathbf{s}^{(i,M)} \sim \mathcal{T}_\Theta(\mathbf{s}, \mathbf{a}, \beta, \hat{\mathbf{q}}, \mathbf{b})$, recording the positions $\mathcal{P}^{(i)} = \{\mathbf{p}^{(i,1)}, \mathbf{p}^{(i,2)}, \ldots, \mathbf{p}^{(i,M)}\}$ and the durations $\mathcal{I}^{(i)} = \{\Delta\tau^{(i,1)}, \Delta\tau^{(i,2)}, \ldots, \Delta\tau^{(i,M)}\}$. We subsequently fit a per-dive Kernel Density Estimator (KDE) for simulator positions with a Gaussian kernel $\hat{p}_\Theta^{(i)} = \text{KDE}(\mathcal{P}^{(i)}; h)$ and bandwidth h, and a Gaussian distribution $\widehat{\Delta\tau}_\Theta^{(i)} \sim \mathcal{N}(\mathcal{I}^{(i)}; \mu^{(i)}, \sigma^2)$ for dive durations with a fixed σ.

Evaluating the probabilities $\hat{p}_\Theta^{(i)}(\mathbf{p}_{\text{true}}^{(i)})$ and $\widehat{\Delta\tau}_\Theta^{(i)}(\Delta\tau_{\text{true}}^{(i)})$ quantifies how well the simulator parameters Θ fit a particular dive. Furthermore, we also penalise the spread $\sum_{j=1}^{M} (\mathbf{p}^{(i,j)} - \overline{\mathbf{p}^{(i,j)}})^2$ of the positions relative to their mean. Without this penalty, the optimisation procedure tends to favour parameter configurations with overly diffuse transition distributions. Better scores are obtained on the worst dives as probability mass is placed on a wide range of outcomes. This behaviour is undesirable, as it leads to uninformative predictions that are also detrimental to the downstream planning. The scoring function $J(\Theta)$ is therefore defined as:

$$J(\Theta) = \sum_{i=1}^{N} \log \hat{p}_\Theta^{(i)}(\mathbf{p}_{\text{true}}^{(i)}) + \lambda_{\text{dur}} \sum_{i=1}^{N} \log \widehat{\Delta\tau}_\Theta^{(i)}(\Delta\tau_{\text{true}}^{(i)}) - \lambda_{\text{reg}} \sum_{i=1}^{N} \sum_{j=1}^{M} (\mathbf{p}^{(i,j)} - \overline{\mathbf{p}^{(i,j)}})^2,$$

where the first term corresponds to the log-likelihood $\ell(\mathcal{D}; \Theta)$ of surfacing locations, the second term corresponds to the duration score with a configurable weight λ_{dur}, and the third term penalises the spread of samples generated for each $\mathbf{x}^{(i)}$ with a configurable weight λ_{reg}.

We seek to find the optimal parameters $\Theta^* = \text{argmax}_\Theta(J(\Theta))$. This high-dimensional optimisation problem has a scoring function that is expensive to evaluate and cannot be computed in closed form. As such, Bayesian optimization is a suitable framework. We opt for BoTorch [32], [33], whose acquisition function navigates the exploration-exploitation tradeoff to generate promising candidate parameter sets Θ_k at each iteration k of a total $n_{\text{SIM-BO}}$ iterations.

E. Sample-Based Online Planner

Gliders surface every few hours and await instructions for the next dive while uploading scientific data. Thus, an *online* planning approach is natural for solving the MDP presented in Section IV.B, leveraging the time on the surface to derive a navigation plan. At each surfacing, the problem instance $I = (\mathbf{p}_0, \tau_0, \mathbf{p}_{\text{goal}}, \rho, \mathbf{q}(\cdot), \mathbf{b}(\cdot))$ is assembled using the position $\mathbf{p}_0$ from the glider's GPS fix, time τ_0, goal specification, current forecasts, and bathymetry.

To perform sample-based online planning and solve the MDP approximately, we leverage Monte Carlo Tree Search (MCTS), an online search algorithm for determining the best action to apply at the current state of an MDP. As a full review of MCTS is out of scope, we refer the interested reader to [26] for more details. Specifically, we opt for the Upper Confidence Bounds for Trees (UCT) [34] variant, which treats action selection at each node as an independent multi-armed bandit problem, applying the principles behind the Upper Confidence Bounds (UCB) [35] algorithm to the tree search setting. Planning is carried out while the glider is at the surface using n_{trials} search tree traversals. n_{trials} is a parameter that is set on a per-deployment basis to account for the expected time duration of the glider at the surface and the computational resources available.

Algorithm 2 Sample-Based Online Planner

Function `PlanNextDive`(I):
  `allActions` $\leftarrow$ [ ]
  for each j out of n_{threads} **in parallel**
    `seed` $\leftarrow$ `SampleRandomState`(j)
    `action` $\leftarrow$ `RunUCT`(I, `seed`)
    `allActions.Append(action)`
  $\mathbf{a} \leftarrow$ `MajorityVote(allActions)`
  return $\mathbf{a}$
Function `RunUCT`(I, `randomState`):
  $(\mathbf{p}_0, \tau_0, \mathbf{p}_{\text{goal}}, \rho, \mathbf{q}(\cdot), \mathbf{b}(\cdot)) \leftarrow I$
  create root node v_{root} from $s_0 = (\mathbf{p}_0, \tau_0)$
  for i in $[1, n_{\text{trials}}]$ **do**
    $v_{\text{leaf}} \leftarrow$ `TreePolicy`(v_{root})
    $\mathbf{s} \leftarrow$ `GetState`(v_{leaf})
    if $\mathbf{s} \in \mathcal{G}$ **then**
      $r \leftarrow$ `SumCosts`($\mathbf{s}_0, \mathbf{s}$)
    else
      $r \leftarrow \epsilon_{\text{heur}} * (d(\mathbf{p}, \mathbf{p}_{\text{goal}})/s)$
    `Backup`(v_{leaf}, r)
  `action` $\leftarrow$ `RobustChild`(v_{root})
  return `action`

Pseudocode for the planner is given in Algorithm 2. We use the following enhancements over the basic MCTS scheme. Our planner leverages *double progressive widening*, which is suitable for scenarios with stochastic transitions and continuous state spaces [36], [37]. It is able to achieve a suitable compromise between bias (induced by the first expanded nodes) and variance (in the result of trials for each state). We note that, despite the discretisation of the actions, we opt for progressive widening in the action space also. This is done to keep the branching factor low and increase the tree depth, enabling us to plan over a longer horizon.

Furthermore, we use *root parallelism* in order to take advantage of multi-core architectures despite the sequential nature of MCTS. This creates n_{threads} search trees that progress independently from the given root state. The final results are aggregated by majority voting to obtain a single action [38], [39]. This improves the robustness of the navigation plan to both noise in the current forecasts and action choices early in the tree search.

We also highlight the following differences from a standard implementation:

- **Use of heuristic.** We do not perform rollouts to estimate the value of leaf nodes as sampling transitions requires running a computationally intensive simulator. This is impractical given the time budget available at each surfacing. Instead, we use a heuristic (line 18) that estimates the time to reach the goal based on the distance to goal and constant speed. The error term ϵ_{heur} of the heuristic was calibrated based on preliminary deployment data and was set to approximately 1.77. Equivalently, the typical trajectory length was $1.77\times$ that of the straight line.
- **Prioritising STG action.** When expanding a node, we always add the child corresponding to the STG action ($\alpha = 0$) first. Further actions are sampled randomly from $\mathcal{A}$ as visit counts increase and the progressive widening condition is triggered. This prioritises an action that leads to good returns in most situations and corresponds to the standard navigation baseline.

V. GLIDER AUTONOMY LONG-TERM PLANNING ENGINE

In this section, we describe the components required for translating our methodological and algorithmic contributions discussed in the previous section to a field-validated system that performs long-term autonomous navigation planning for underwater gliders. The *Glider Autonomy Long-term Planning Engine* was designed and implemented in collaboration with operational stakeholders in order to enhance operational practicality and safety.

A. Automated Goal Management

The previous section specified problem instances I with a single goal $\mathbf{p}_{\text{goal}}$. However, glider missions typically consist of many such goals. A common use case, which we target, is to have the glider sample data between a set of predefined points, i.e., a *transect* in the case of two points, and the corners of a polygon more generally. We therefore assume we are given an ordered list of points $\{\mathbf{p}_{\text{goal}}^{(i)}\}_{i=1}^{n_{\text{goals}}}$. The system advances to the next goal in the list upon achieving the current one, and cycles to the beginning of the list once the end is reached.

For determining whether the goal was achieved, i.e., whether $\mathbf{s} \in \mathcal{G}$, an additional condition must be checked in GALE besides surfacing within the goal region. A goal may also be detected as having been achieved while underwater using dead reckoning (which we note may be inaccurate). This reflects the operation of the on-board glider software.

B. Translation of Control Instructions

Actions $\mathbf{a} = (\alpha, \mathbf{c})$ output by the planner must be translated to *files* that can be understood by a Slocum glider. Two types of files are required: a *waypoint list* specifying an ordered sequence of coordinates that the glider should reach, and an *instruction set* containing the control parameters $\mathbf{c}$.

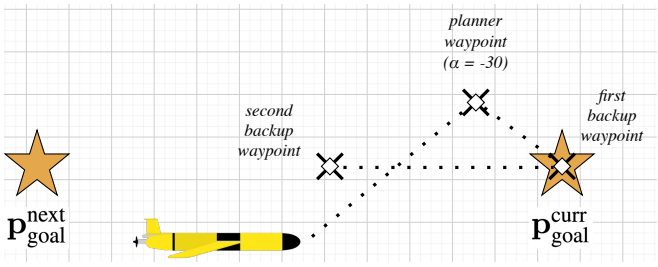

FIGURE 7: Visualisation of waypoint list generation. The first waypoint corresponds to the planner action (here, $\alpha = -30$). The first backup waypoint is set to be the current goal as otherwise the goal would be overshot. The second backup waypoint is aimed straight at the next goal.

To generate the waypoint list, we use the α component of the planner action for computing the first waypoint and several backup waypoints derived using straight-to-goal bearings. Since waypoint lists are recomputed and updated at each surfacing, the vehicle can steer according to planner output during normal operation and use a sensible backup trajectory otherwise. As mentioned in Section II, several failures in transferring control instructions can occur in a row. Therefore, waypoint lists must extend well beyond a single dive so that the vehicle can continue its mission uninterrupted in the case of communication failures.

A waypoint list generation example is shown in Figure 7. The algorithm, which we describe at a high level due to space limitations, takes as input the action α, glider position $\mathbf{p}_0$, the position of the current goal $\mathbf{p}_{\text{goal}}^{\text{curr}}$, the position of the next goal $\mathbf{p}_{\text{goal}}^{\text{next}}$, the goal tolerance radius ρ, the waypoint distance ρ_{wpt}, and number of backup waypoints n_{bck}. The algorithm first determines the waypoint corresponding to the planner action. If distance to the goal is low, the goal itself is used as the waypoint to avoid oscillating around it. Otherwise, the waypoint is set along the heading of the planner action at the specified waypoint distance. Backup waypoints towards the current goal and, if required, the next goal are added. The bearings for the backup waypoints are computed starting from the previous waypoint. Additional backup waypoints towards a particular goal are not necessary if the previous waypoint is already within the same goal region. If a backup waypoint overshoots the goal region, it is replaced with the goal itself.

Generating the instruction set file involves a format translation of the control parameters $\mathbf{c}$. Following the guidance of glider pilots, GALE supports specifying a number of fixed instruction sets $\mathcal{K}$ from which $\mathbf{c}$ is chosen when the mission is initialised. This ensures the control parameters are within a field-tested range of configurations, protecting the assets and ensuring pilot trust in the system. As previously mentioned, for simplicity, we assume that a fixed $\mathbf{c}$ is used for the duration of a particular mission. This assumption is realistic for the deployments considered in our evaluation,

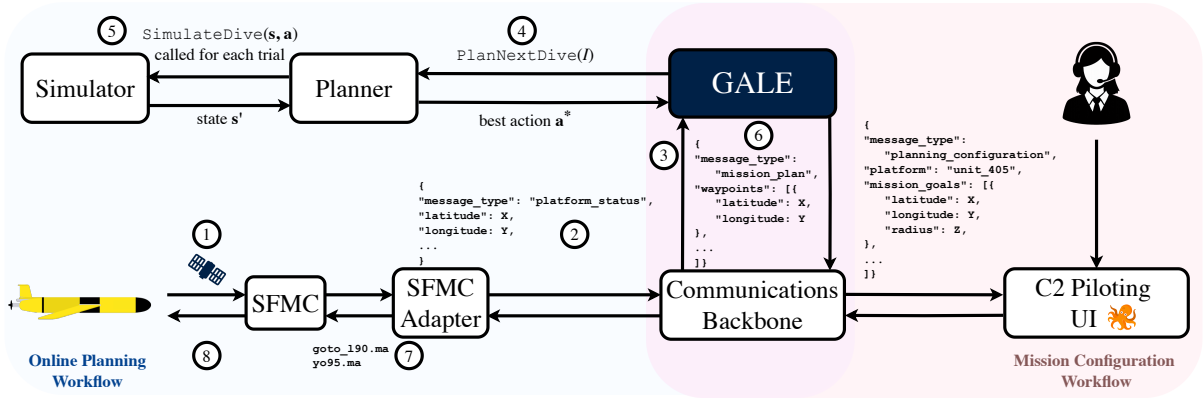

FIGURE 8: Diagram illustrating the components and steps of the two system workflows. The Mission Configuration Workflow (right) is executed once at the start the mission, while the Online Planning Workflow (left) is executed every few hours upon the surfacing of the glider.

which require sampling of scientific data at regular time and distance intervals. Dynamically varying the instruction set on a per-action basis is left for future work.

C. System Components and Integration

The GALE system implements the methodology described in Section IV and is a service deployed to an on-premise server. It is capable of controlling several gliders simultaneously and fetches the most recent current forecasts daily. In this section, we describe the components that are required besides GALE to achieve the full control loop of the glider.

C2 Communications Backbone. This open-source component [40], [41] developed by the UK National Oceanography Centre (NOC) enables real-time transfer of vehicle information and control instructions using standardised APIs [42].

SFMC. Slocum Fleet Mission Control (SFMC) provides interfaces for controlling gliders manually or programmatically. It enables transferring files containing waypoints and control parameters by placing them in a dedicated directory from which they are transmitted via an Iridium satellite link.

C2 Piloting User Interface. This web-based interface supports configuring missions for, monitoring, and piloting marine autonomous systems. It is complementary to SFMC and unifies piloting interfaces for several platforms [40].

D. Workflows

Two workflows are illustrated in Figure 8.

Mission Configuration Workflow. This workflow is executed once at the start of a mission. It involves the use of the C2 Piloting UI to indicate the glider, the list of goals $\{\mathbf{p}_{\text{goal}}^{(i)}\}$, and permitted instruction sets $\mathcal{K}$. The GALE system initialises the tracking information in persistent storage. From this point onwards, GALE will track the mission goals and respond to post-surfacing updates from the glider with navigation plans.

Online Planning Workflow. This workflow achieves the full glider control loop. Upon surfacing, the glider calls in to SFMC (1). Over satellite, it relays status information (time, GPS fix) and transfers scientific data while awaiting instructions from GALE, which creates a corresponding problem instance I (2, 3). The Planner (Algorithm 2) is invoked using I, which calls the Simulator (Algorithm 1) on the order of $10^3 - 10^4$ times in each thread (4,5). The best action $\mathbf{a}$ is determined by the Planner and converted to a waypoint list and instruction set, which are published to the Backbone (6). The messages are placed as files in the appropriate directory on SFMC (7). They are transmitted to the glider and are used to execute the next dive (8).

It is worth reiterating that steps 1 and 8, which involve communication over satellite, are not fully reliable. In preliminary deployments, the rate of successful transfer of waypoint lists and instruction sets was approximately 89%, with further significant drops during periods with adverse weather conditions. Since navigation plans cover a longer horizon than individual dives (on the order of a day rather than hours), the glider was able to continue its mission.

E. System Configuration

Configuring the system for a new deployment requires the following steps:

- The scientific stakeholders supply the mission goals $\{\mathbf{p}_{\text{goal}}^{(i)}\}$ and the goal tolerance radius ρ.
- The glider pilots specify the permitted instruction sets $\mathcal{K}$ that are appropriate for the chosen vehicle and the scientific objectives.
- Relevant data from historical glider deployments is identified and retrieved (e.g., from public repositories [43]). Alignment to the upcoming deployment in terms of location, vehicle type, and time period are important for the faithfulness of the simulator.

TABLE 1: Deployment goal locations and parameters. All distances are given in km.

Deployment	Goal 1	Goal 2	Dist.	ρ	ρ_{wpt}	n_{bck}
MOGli	59.32°N 0.35°W	59.32°N 0.65°W	17.0	2.0	7.0	2
eSWEETS^3 N	57.089°N 1.856°W	57.062°N 1.698°W	10.0	1.0	7.0	2
eSWEETS^3 S	56.929°N 1.945°W	56.903°N 1.787°W	10.0	1.0	7.0	2

TABLE 2: Deployment summary statistics.

Deployment	Transects completed	Duration (days)	Total dives	Distance travelled (km)
MOGli	41	42	343	665
eSWEETS^3 North	20	27	150	173
eSWEETS^3 South	23	22	147	207

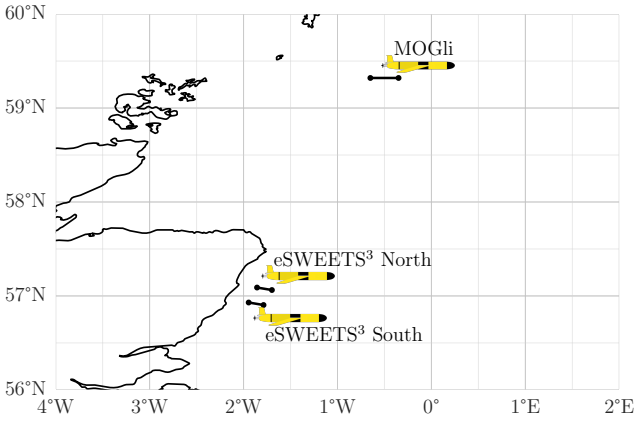

FIGURE 9: Field deployment locations in the North Sea.

- The GALE system administrator performs simulator parameter tuning (Section IV.D) using the historical data. Planner parameter tuning is also performed given the tuned parameters for the simulator. An instance of the GALE system is configured and deployed to computational infrastructure.
- The vehicle is deployed and piloted manually towards the operational area of the campaign.
- The pilots execute the Mission Configuration Workflow (Section V.D). This signals the start of the automated piloting, completing the setup for this deployment.

VI. EXPERIMENTAL SETUP

A. Field Experiments

Our system was evaluated in two observational campaigns in the North Sea: *MOGli* (one glider) and *eSWEETS^3* (two gliders simultaneously). In these deployments, gliders were operated by NOC on behalf of external partners the UK Met Office and the University of East Anglia. A transect formed of two goal positions was used in all deployments. Deployment coordinates and summary statistics are given in Tables 1 and 2. A map of the three field deployments is shown in Figure 9.

MOGli. *Met Office Glider (MOGli)* is an ongoing joint project between the UK Met Office (MO) and NOC. Measurements of ocean properties are gathered by a single glider and transmitted periodically to the MO. This data is used to refine MO forecast models. The use of GALE to pilot the glider leads to a virtuous circle as AMM15 forecasts are used to derive navigation plans, while the scientific data itself is used to improve the forecasts.

The glider was deployed in a region with low bottom temperature variance since this was part of the MOGli scientific objectives. Fully autonomous control operated from 2 May to 3 July 2025, following preliminary experiments that began in November 2024. A pause in autonomous control was in place from 21 May to 7 June 2025 while the glider was recovered, recharged, and redeployed to the region.

The action space was chosen to contain relative bearings $\alpha \in \{-40, -20, 0, 20, 40\}$. For the fixed control parameters $\mathbf{c}$, a value of $n_{\text{yos}} = 5$ yos was chosen to lead to an operationally appropriate dive duration of approximately 3 hours. A target depth of $z_{\text{bottom}} = 150$ was used, which exceeds the depth of the seabed in this region. This was driven by the requirement to sample the entire water column, and hence the glider uses the altimeter to inflect when close to the seabed, a feature also supported by our simulator.

eSWEETS^3. In the *enabling Sustainable Wind Energy Expansion in Seasonally Stratified Seas (eSWEETS^3)* project, two Slocum gliders were deployed close to the Kincardine floating offshore wind farm. The gliders were positioned upstream (referred to as eSWEETS^3 North) and downstream (eSWEETS^3 South) of the wind farm to assess the impacts of the floating structures on the local turbulent mixing, which in turn influences oxygen availability, nutrient supplies, and biological productivity.

The project deployments were performed in two phases. In the first phase (May - August 2025), gliders were piloted manually using standard procedures. In the second phase (August - October 2025), GALE was used to control the gliders following preliminary experiments. Fully autonomous operation was in place from 2 September to 30 October 2025 for eSWEETS^3 North and from 18 September to 30 October 2025 for eSWEETS^3 South.

The chosen action space contained relative bearings $\alpha \in \{-90, -60, -30, 0, 30, 60, 90\}$, a wider range than in MOGli to account for the effect of strong tidal currents, which were aligned roughly orthogonal to the glider transects. $n_{\text{yos}} = 10$ yos were used, leading to dives lasting approximately 4 hours in this shallower region. A target depth of $z_{\text{bottom}} = 150$ was used which, as in MOGli, exceeds the bathymetric limit.

The eSWEETS^3 deployments were subject to additional operational requirements. As the gliders were deployed in

areas with strong tidal currents, there was a risk of vehicle loss if it drifted away from the mission area, especially into the wind farm or shipping lanes. *Safety polygons* were defined, and additional functionality was implemented such that, if the glider drifted outside of the safety polygon, the algorithm described in Section V.B is replaced with a simplified navigation plan aiming directly for the center of the polygon. Normal navigation was resumed upon re-entry. Furthermore, as continuing to sample along the transect upon re-entry was deemed more important for the scientific objectives than turning back to achieve the goal, the system switched to the next goal if the glider had already progressed past 90% of the transect before exiting the safety polygon.

Baseline. As a baseline for the Planner, we consider the *Straight-to-Goal* approach, which always selects the action with $\alpha = 0$. For an equitable comparison, the glider remains under the autonomous control of GALE when Straight-to-Goal is used. Since only one of the two methods can control the glider at any one time, an alternating-control approach was taken. The two methods took turns to control the glider after the achievement of 2 goals (equivalently, the algorithm was switched after every back-and-forth transect traversal). Even though the currents experienced by the two algorithms differ, we expect that any bias thus introduced is removed in expectation.

For MOGli, alternating-control was followed throughout. For eSWEETS^3, exclusive use of Straight-to-Goal was mandated by operational concerns until 18 September for eSWEETS^3 North and 2 October for eSWEETS^3 South. GALE was subsequently set to issue Planner waypoints for a comparable duration (until 14 October for eSWEETS^3 North and 15 October for eSWEETS^3 South). For the final period, the algorithms were switched to alternating-control after every back-and-forth transect traversal as in MOGli.

B. Simulation Experiments

We present a set of experiments in simulation whose setup mirrors the deployments listed above. They are an idealised version of the real-world experiments, wherein file transfers always succeed and currents experienced are well-represented by the forecasts. Underwater goal achievement detection (Section V.A) and the additional functionality for eSWEETS^3 (Section VI.A) are not supported in simulation. The simulator and planner parameters are otherwise set identically to those used in the field deployments.

We select three scenarios for each deployment to be replayed in simulation out of the tens of transects completed in the real world (see counts in Table 2). A scenario consists of a starting state (glider position and time) and goal assigned to the glider, which implies a set of historical current forecasts corresponding to the starting state. The same scenarios can be ran as many times as desired with different realisations of the uncertainty, allowing statistically robust comparisons. The baseline remains the same (Straight-to-Goal). In contrast with the field experiments, in simulation the two methods can be compared using identical conditions.

To determine the scenarios, we first assess the number of dives required by Straight-to-Goal to complete each transect averaged over 200 repetitions. We pick the starting conditions for the transects with the minimum (*Favourable*), median (*Neutral*), and maximum (*Unfavourable*) dives required on average for the baseline to reach the goal. This yields scenarios that represent a range of conditions experienced by the glider during the deployment periods. We then re-run both the Planner and Straight-to-Goal over 1000 random seeds to report the final results.

C. Metrics

Planner. To evaluate the planner, we report the total transect duration (i.e., the cumulative cost as defined in Section IV.B), which is the objective that the method optimises for. As secondary metrics, we report the number of dives (highly correlated with total duration) and the total distance covered for each traversal. All ± entries in tables denote 95% confidence intervals. Occasionally, navigation plans were not generated or followed by the glider due to planned (e.g., software upgrades) or unplanned (e.g., anomalies causing abort of the dive) concerns. Even though these events typically impacted a single surfacing, they compromise the transect validity. Such transects were excluded from the planner evaluation.

Simulator. To evaluate the simulator, we report the true and simulated dive distances and durations as well as their differences (deltas). Since datasets contain different numbers of dives, we report the mean (per-dive) log-likelihood $\bar{\ell}$ instead of the summed log-likelihood ℓ. These metrics are assessed on the dives recorded during the live deployment, which were not used for simulator optimisation or validation. Furthermore, as the simulator is stochastic and produces several plausible next states, we report statistics with respect to the sample means obtained by drawing 40 samples for each dive.

D. Simulator and Planner Parameter Optimisation

To optimise the simulator parameters (Section IV.D), we use historical glider data and current forecasts gathered in preliminary deployments (concretely, February - March 2025 for MOGli and May - June for eSWEETS^3). The dataset for simulator parameter optimisation is pooled across the two eSWEETS^3 gliders given the proximity of the two regions. Therefore, the same simulator parameters are used in both eSWEETS^3 North and South. We use AMM15 forecasts archived by our system for simulator parameter optimisation. However, such data can also be obtained through publicly available oceanographic data repositories (e.g., [43] and [44]).

The dives are split randomly into training and validation sets. Parameters are optimised using Bayesian optimisation (BO) on the training set, and the best parameters on the validation set with respect to $J(\Theta)$ are selected. We apply

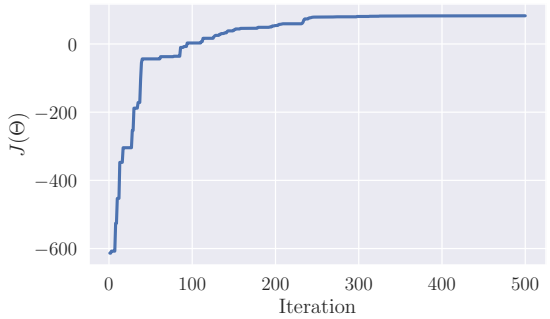

FIGURE 10: Learning curve showing the best value of $J(\Theta)$ found as a function of Bayesian optimisation iterations using the MOGli training data.

TABLE 3: Comparison of simulated and true dive metrics averaged over all dives in the real-world deployments.

Metric	MOGli	eSWEETS^3 North	eSWEETS^3 South
True dive distance (m)	2733	5650	4679
Simulated dive distance (m)	2119	4991	4362
Distance delta (m)	1448	1555	1080
True dive duration (s)	10421	15923	12935
Duration delta (s)	345	1481	1377
Mean log-likelihood $\bar{\ell}$	-9.29	-78.67	-133.90

several data cleaning steps by excluding dives that did not terminate due to normal surfacing, those with speed 2 standard deviations outside the mean, and with a difference between heading and surface location greater than a 75 degree threshold for MOGli and 80 for eSWEETS^3. These criteria resulted in excluding 19.58% of dives for MOGLi and 5.30% dives for eSWEETS^3. $n_{\text{SIM-BO}} = 500$ iterations of BO were used as the scoring function plateaued before this step in all experiments.

In terms of planner parameters, we use $n_{\text{threads}} = 8$ parallel search trees for both deployments. We set $n_{\text{trials}} = 10000$ and $n_{\text{trials}} = 5000$ for MOGli and eSWEETS^3 respectively, as for the latter deployment less data is transmitted and therefore less online planning time is available. We found that the planner hyperparameters (exploration constant, progressive widening parameters, maximum tree depth) are sensitive to simulator parameters. Therefore, after fixing simulator parameters, planner hyperparameters are optimised downstream with BO under the objective of minimising the cumulative cost to reach the goal.

VII. RESULTS

A. Simulator Evaluation

Figure 10 shows the progress of the scoring function $J(\Theta)$ during the steps of the Bayesian optimisation procedure on the MOGli data. Additionally, in Figure 11, we show snapshots of simulator outputs for a MOGli dive at the beginning, intermediate, and final steps of the procedure. The refinement of the simulator output distribution with respect to the true surfacing location is evident.

Several values for the spread penalty λ_{reg} were tested, and the parameters that obtained the best $J(\Theta)$ on the validation set were chosen ($\lambda_{\text{reg}} = 1$ for MOGli, corresponding to the figures shown, and $\lambda_{\text{reg}} = 0$ for eSWEETS^3). The dive durations can realistically be treated as approximately constant in the MOGli region given the assumption of fixed control parameters and the fact that the bathymetry is consistent throughout the transect. Therefore, the duration score coefficient for MOGli is $\lambda_{\text{dur}} = 0$. In contrast, we set $\lambda_{\text{dur}} = 1$ in the eSWEETS^3 region as the eastern ends of the eSWEETS^3 transects are $20 - 40$ m deeper than the western ends. The variation in bathymetry induces variation in dive durations given fixed dive control parameters (i.e., dive durations increase in deeper regions). The standard deviation $\sigma = 700$ was estimated based on data from the first phase which used manual piloting (Section VI.A).

In Table 3, we report several metrics concerning simulator performance on the deployment data (not seen during training or model selection). The distances of the dives output by the simulator were close to their true average on eSWEETS^3, but noticeably underestimated for MOGli. The distance between the true and simulated surfacing locations is around 1.5 km on average for all deployments (though it is inherently challenging to draw conclusions by comparing an individual sample with the empirical mean of the simulator distribution). In contrast, the dive durations were more accurate for MOGli given the aforementioned variation in bathymetry for eSWEETS^3. The mean log-likelihood $\bar{\ell}$ degraded on the deployment data compared to the training data, as would be expected. On the deployment data, the optimised simulator parameters yielded scores of $\bar{\ell} = -9.29$ for MOGli and $\bar{\ell} = -106.00$ for eSWEETS^3, which are significantly better than those of randomly chosen parameters at the start of optimisation ($\bar{\ell} \approx -32$ and $\bar{\ell} \approx -285$ respectively). This evidences the benefits of the proposed tuning procedure. The log-likelihood scores are worse for eSWEETS^3 compared to MOGli due to the stronger currents in this tidal region.

We note that there are many possible causes for predictive errors in the simulator, which include the following: forecasts being wrong, the steady-state approximation of our simulator, mismatches between the physical glider used for simulator optimisation and the one used in the real world (distinct vehicles with variations in payload and ballasting), and noise in how the glider executes the control commands. It is highly likely that all these factors are at play and, while simulator accuracy can be improved, some degree of error is unavoidable.

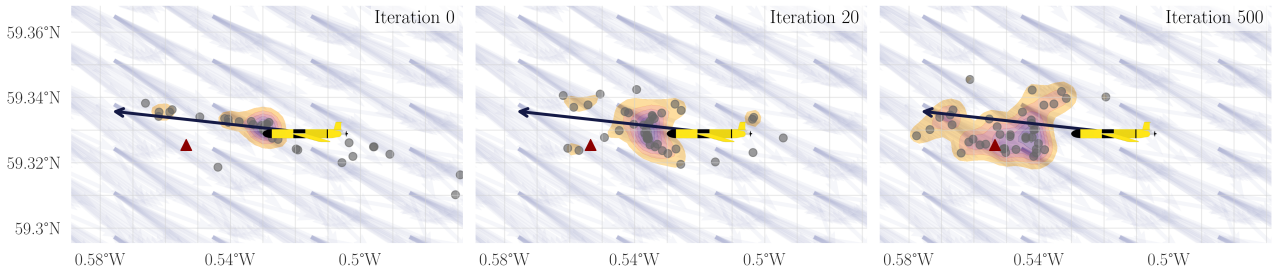

FIGURE 11: Progress of the proposed simulator parameter optimisation procedure (Section IV.D) on one of the dives in the MOGli deployment. Simulator parameters are progressively adjusted such that the surfacing locations generated by the simulator (grey circles) best match the true surfacing location (red triangle).

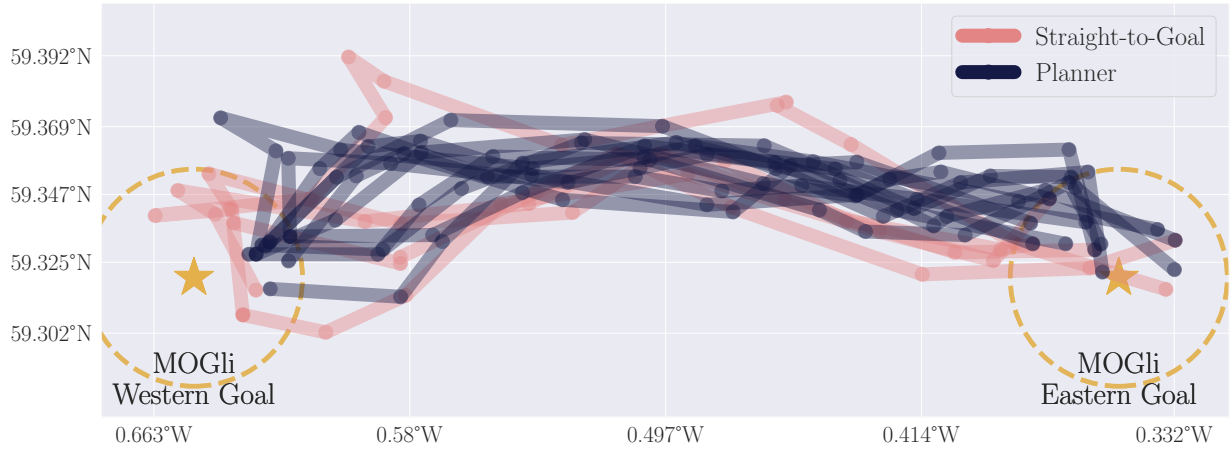

FIGURE 12: Transects completed by the proposed GALE system in the MOGli field deployment using the Planner and the Straight-to-Goal baseline. Only transects in which all transfers of control instructions succeeded are shown. The goal radii are represented by two dotted circles around the goals. The Planner results in transects with shorter minimum, mean, and maximum durations. Transects produced by the two methods are also discernible visually. The transects completed by the Planner have statistically significantly shorter path lengths and avoid overshooting the western goal.

B. Planner Evaluation in Simulation

Table 4 shows the results comparing the Straight-to-Goal and Planner algorithms in simulation. The Planner improves over Straight-to-Goal in duration and path length in all settings tested. The most significant improvement is obtained for the Favourable scenario in the eSWEETS[3] North deployment, in which the planner reduces the mean transect duration by 7 hours and the path length by 11 km. For MOGli, the reductions in duration over the baseline are small in an absolute sense at slightly over 30 minutes per transect, equating to a 2.39% decrease. This region does not exhibit strong currents, as corroborated by the small differences in the performance of STG between the most Favourable and Unfavourable scenarios. Therefore, our currents-aware approach does not have much room for improvement over the STG algorithm. The improvements are more pronounced for eSWEETS[3], where the Planner leads to decreases of 9.88% and 8.92% respectively in dive duration. The reductions in path length of $4.23 - 13.94\%$ are more pronounced than the duration reductions. This is an important secondary metric as sampling along trajectories that are as straight as possible facilitates robust environmental data analysis and helps achieve many scientific objectives.

C. Planner Evaluation in Field Deployments

Results for the real-world deployments are reported in Table 5. These show that the Planner outperformed the baseline in 2 out of 3 deployments. For MOGli, as in simulation, the Planner leads to a reduction in the mean duration of approximately 30 minutes on average; the minimum and maximum durations across all transects are also reduced. The Planner yields a reduction in the path length that is statistically significant at the 5% level ($p = 0.041$ and rank-biserial $r = 0.376$ under the nonparametric Mann-Whitney U test with 20 samples for STG and 21 for the Planner). However, given the number of transects recorded (see Table 2), statistical significance was not obtained for other deployments and metrics. These field results should

TABLE 4: Results comparing the Straight-to-Goal and Planner algorithms across several scenarios in simulation. Bold indicates the better-performing algorithm.

Deployment	Scenario	Algorithm	Mean Duration (hrs)	Mean Number Dives	Mean Path Length (km)
MOGli	Fav.	STG	22.48±0.30	7.68±0.10	17.31±0.05
		Planner	**22.13**±0.30	**7.57**±0.10	**16.79**±0.06
	Neutral	STG	22.93±0.31	7.87±0.11	17.66±0.05
		Planner	**22.38**±0.30	**7.68**±0.10	**16.87**±0.06
	Unfav.	STG	23.60±0.34	8.10±0.12	17.85±0.06
		Planner	**22.84**±0.32	**7.80**±0.11	**16.92**±0.06
eSWEETS^3 North	Fav.	STG	55.67±1.40	10.01±0.26	52.31±1.54
		Planner	**48.70**±1.44	**8.96**±0.28	**41.10**±1.45
	Neutral	STG	65.70±0.87	11.33±0.16	52.92±1.15
		Planner	**62.91**±0.92	**10.87**±0.17	**46.67**±1.06
	Unfav.	STG	64.07±1.18	11.79±0.22	65.20±1.29
		Planner	**55.82**±1.43	**10.24**±0.27	**54.57**±1.40
eSWEETS^3 South	Fav.	STG	43.46±1.47	**9.70**±0.36	43.60±1.58
		Planner	**41.81**±1.50	9.78±0.40	**40.09**±1.50
	Neutral	STG	55.63±1.14	11.80±0.28	53.79±1.23
		Planner	**47.29**±0.98	**10.08**±0.25	**43.09**±0.95
	Unfav.	STG	61.99±0.91	13.04±0.22	68.77±1.00
		Planner	**57.05**±1.11	**12.52**±0.29	**59.21**±1.12

be treated as corroborating evidence alongside the larger-scale simulation study rather than as standalone proof of generality. The Planner also performed better than STG for eSWEETS^3 North in all metrics. For eSWEETS^3 South, the mean values were slightly worse, while the minimum / maximum ranges still matched or improved over STG.

In Figure 13, we analyse the frequency of the actions taken by the Planner in the three deployments. While the STG action ($\alpha = 0$) is the most selected overall, the Planner often chooses to steer relative to the goal. In the eSWEETS^3 region, characterised by stronger tidal currents, the distribution of chosen actions is wider and flatter, and even the most extreme actions are still chosen 15% of the time in eSWEETS^3 North. The strong tidal currents in the eSWEETS^3 region also resulted in surfacings outside of the safety polygon. 27 such surfacings (20.3%) occurred for eSWEETS^3 North and 4 (2.9%) for eSWEETS^3 South.

D. Alignment of Simulation and Real-World Results

For MOGli, the duration and number of dives are mostly aligned, while the path lengths are noticeably shorter in simulation. This is consistent with the distance underestimation observed in the simulator evaluation. For the eSWEETS^3 deployments, values for all metrics were consistently higher in simulation compared to the field deployments. Given the values reported in Section VII.A, simulator predictions of dive distance and durations are well-aligned with field observations. Consequently, the difference in transect metrics is attributable to the goal achievement conditions in the deployed autonomous system being broader (3 conditions, as described in Sections V.A and VI.A) than those in simulation

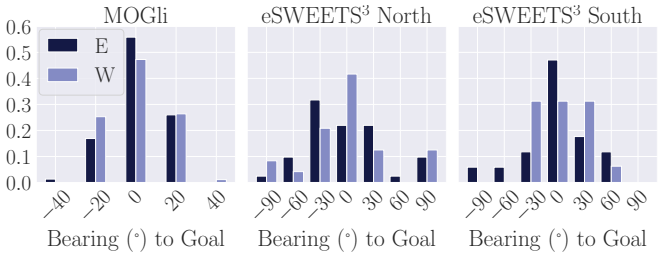

FIGURE 13: Proportions of actions chosen by the Planner during the three field deployments, split between the East and West directions of travel. The stronger currents of the eSWEETS^3 region resulted in a wider distribution of action frequencies.

(1 condition). Requiring surfacing within a tight radius in simulation will result in overshooting the goal and hence longer transects until the goal is eventually achieved. Concretely, in the eSWEETS^3 field trials, the goal was deemed achieved by reaching it underwater in 46% of cases, having traversed 90% of the transect length prior to safety polygon re-entry in 22% of cases, and by surfacing within the goal radius ρ in only 32% of transects.

E. Summary of Results

Overall, our results demonstrate that the proposed online planning techniques outperform the standard Straight-to-Goal baseline. Improvements of up to 9.88% in dive duration and 16.51% in path lengths were obtained in simulations calibrated on real-world data. Our system also obtained a statistically significant path length reduction of 9.55% in a field deployment. Compared to prior studies that are limited to simulation or short-term field trials, these results provide evidence of performance gains in real-world operational campaigns. A key achievement of our work is the demonstration of sustained autonomous operation at scale. The GALE system was deployed in two operational campaigns totalling over 3 months and 1000 km of autonomous glider operation, successfully addressing the challenges of environmental and actuation uncertainty, intermittent communication, and long-horizon navigation planning.

VIII. DISCUSSION

A. Lessons Learned and Engineering Considerations

The feedback of pilots involved in the two campaigns highlighted the value of the system in automating standard operation. The system reduced manual effort and allowed pilots to focus on monitoring and non-routine interventions for other campaigns. In the eSWEETS^3 campaign, the safety polygon feature allowed for automated intervention when pilot supervision may not have been available otherwise. Additionally, the straighter transects attained by our planning approach lead to more consistent sampling along predefined

TABLE 5: Results comparing the Straight-to-Goal and Planner algorithms in the field deployments. Bold denotes the better-performing algorithm for each deployment and metric.

Deployment	Algorithm	Mean Duration (hrs)	Min Duration (hrs)	Max Duration (hrs)	Mean Number Dives	Min Number Dives	Max Number Dives	Mean Path Length (km)
MOGli	Straight-to-Goal	23.09	16.00	31.70	8.45	6	11	22.58
	Planner	**22.68**	**14.35**	**26.98**	**8.28**	**5**	**10**	**20.52**
eSWEETS^3 North	Straight-to-Goal	40.38	19.80	83.54	9.43	5	18	49.10
	Planner	**28.88**	**18.05**	**43.10**	**6.46**	**4**	**10**	**38.36**
eSWEETS^3 South	Straight-to-Goal	**21.37**	10.75	54.64	**6.00**	3	14	**29.29**
	Planner	26.82	**10.44**	**47.55**	7.29	3	**13**	32.95

paths, which supports robust environmental data analysis and aligns with common scientific objectives.

The early stages of the project built gradually towards achieving the full control loop. For testing the systems integration, a "shadow" glider was initially configured. This glider duplicated the status updates received from the real glider. The control instructions generated by GALE were placed in a special folder where they could be sanity-checked by pilots but not acted upon. Subsequently, pilots began transferring the validated control instructions manually to the real glider. Once confidence in the system was achieved, fully autonomous piloting by GALE was enabled during working hours, with manual piloting outside of this window. By the start of the MOGli deployment, pilots were comfortable with simply monitoring the operation of the fully autonomous system regularly.

Regarding the generality of our system, given the modular architecture, GALE is extensible to other glider designs such as the Seaglider [45]. From a software engineering standpoint, this would require replacing the SFMC and SFMC Adapter with vehicle-specific piloting systems and adapters. From a methodological point of view, while Seagliders have slightly different low-level actuators, they also require the specification of waypoint and control parameters in an online piloting loop. Thus, our methodology is applicable to other glider designs with minimal modifications.

B. Challenges and Future Work

We aim to generalise our method to also consider the optimisation of energy use, which is a feasible extension of the proposed formulation. As the duration and energy use objectives conflict, a multi-objective approach [46] is required. This would allow the selection of more energy-intensive controls during periods with strong currents in order to make sufficient progress towards the scientific objectives. Extending the method to incorporate constraints and guarantees, such as staying within a specified survey area or avoiding certain areas (e.g., shipping lanes) would broaden the applicability of the method. The problem may also be framed as a POMDP [47] in order to maintain beliefs about the nature of forecast errors, which would require in-situ current measurements.

Replacing or augmenting the simulator with a learned model, which has proven successful with a variety of planning and reinforcement learning approaches [48], [49], may speed up execution and thus enable planning over a longer horizon. As shown by our simulator evaluation, further improvements in accuracy are possible. However, fitting simulator behaviours using different vehicles and deployment environments is a significant challenge due to distribution shifts. Further developments to the planner can include the aggregation of continuous actions for root-parallel MCTS [50], improving the exploration strategy [51], and contextually deciding which actions to prioritise.

Evaluating glider navigation planning algorithms in the field is inherently challenging given the long traversals and, consequently, the low number of transects. If resources permit, running algorithms on gliders deployed side-by-side in a paired experimental design would reduce variability and temporal dependencies compared to the alternative-control design we adopted. Regardless, replaying on-field conditions in suitably calibrated simulators, as performed in our evaluation, is ultimately necessary for statistically robust findings.

Failures in transferring instructions to the glider also pose a unique challenge, as they can cause the glider to turn around towards a stale waypoint. Such failures disproportionately impact navigation plans with waypoints that steer away from the goal to compensate for currents. This phenomenon highlights the trade-off between the gains of online planning for steering given up-to-date current forecasts versus the negative impacts of using a stale online plan due to failed instruction transfers. This can be addressed by explicitly reasoning about the time available for planning [52] and limiting it in order to improve the probability of instruction transfers succeeding.

In terms of applications, we envisage leveraging the present work as a building block. The system can operate in regions with high eddy activity using high-resolution ocean forecast models, a task that is challenging for current piloting methods. Identifying ocean fronts from horizontal temperature gradients [53] or tracking algal blooms [54] can be achieved by creating a higher-level component that reasons about where the highest-value observation(s) can be

gathered and tasking the glider accordingly. Decision-making could also be implemented to minimise biases identified in existing sampling strategies (e.g., [55]). For going beyond the single-glider scenario, coordination strategies can be considered in a multi-robot setting. High-level methods can be constructed that assign goals or regions to individual gliders, with our planner being used to compute uncertainty-aware navigation plans for each vehicle.

IX. CONCLUSION

In this paper, we have considered the problem of navigation planning for underwater gliders, a type of robot that has become invaluable for ocean sampling. We have taken important steps to address the current methodological and systems gaps towards achieving long-term autonomous glider operation. Our work proposed a glider simulator and a procedure for adjusting its parameters using historical dives performed in the real world. We have also contributed a sample-based online planning technique for deciding how to control the glider with the objective of minimising duration to reach the goal. Lastly, in collaboration with pilots and domain experts, we have designed the Glider Autonomy Long-term Planning Engine, which is suitable for long-term and multi-platform missions.

We have thoroughly validated our approach in simulation and through operational campaigns in the North Sea, in which our system was used to control gliders collecting data towards refining weather models and understanding the impacts of offshore wind farms. Our work improves the sampling of scientific data and paves the way towards the autonomous management of fleets of gliders at scale, reducing costs and allowing human pilots to focus on monitoring and non-routine interventions. Our contributions ultimately support advancements in marine science and policy.

ACKNOWLEDGMENTS

This work was supported by the UK Natural Environment Research Council (NERC) under the UKRI TWINE programme (grant number NE/Z503381/1) and the Doctoral Training Partnership in Environmental Research (NE/S007474/1). It was also supported by the UK Engineering and Physical Sciences Research Council (EPSRC) From Sensing to Collaboration Programme Grant (EP/V000748/1) and the Innovate UK AutoInspect Grant (1004416). The authors would like to gratefully acknowledge the contribution of the UK Met Office and the University of East Anglia, who gave permission for the GALE system to be trialled in their glider campaigns. The eSWEETS[3] glider campaign was funded by the UK NERC (grant number NE/X004872/1). Lastly, we acknowledge the use of the University of Oxford Advanced Research Computing (ARC) facility in carrying out this work (see http://dx.doi.org/10.5281/zenodo.22558).

ADDITIONAL INFORMATION

A patent application has been submitted on the work covered in this paper, which was first filed on 19 December 2024 under UK application number GB2418678.5.

REFERENCES

[1] D. L. Rudnick, R. E. Davis, C. C. Eriksen, D. M. Fratantoni, and M. J. Perry, "Underwater Gliders for Ocean Research," *Marine Technology Society Journal*, vol. 38, no. 2, pp. 73–84, 2004.

[2] D. Roemmich, S. Riser, R. Davis, and Y. Desaubies, "Autonomous Profiling Floats: Workhorse for Broad-scale Ocean Observations," *Marine Technology Society Journal*, vol. 38, no. 2, pp. 21–29, 2004.

[3] D. Roemmich, "On the Future of Argo: A Global, Full-Depth, Multi-Disciplinary Array," *Frontiers in Marine Science*, 2019.

[4] National Oceanography Centre, "Net Zero Oceanographic Capability (NZOC) summary report," https://noc.ac.uk/files/documents/facilities/NZOCSUMMARYREPORTV2.pdf, 2021, accessed: 2026-03-31.

[5] P. Testor, B. De Young, D. L. Rudnick, S. Glenn, D. Hayes, C. M. Lee, C. Pattiaratchi, K. Hill, E. Heslop, V. Turpin, P. Alenius, C. Barrera, J. A. Barth, N. Beaird, G. Bécu, A. Bosse, F. Bourrin, J. A. Brearley, Y. Chao, S. Chen, J. Chiggiato, L. Coppola, R. Crout, J. Cummings, B. Curry, R. Curry, R. Davis, K. Desai, S. DiMarco, C. Edwards, S. Fielding, I. Fer, E. Frajka-Williams, H. Gildor, G. Goni, D. Gutierrez, P. Haugan, D. Hebert, J. Heiderich, S. Henson, K. Heywood, P. Hogan, L. Houpert, S. Huh, M. E. Inall, M. Ishii, S.-i. Ito, S. Itoh, S. Jan, J. Kaiser, J. Karstensen, B. Kirkpatrick, J. Klymak, J. Kohut, G. Krahmann, M. Krug, S. McClatchie, F. Marin, E. Mauri, A. Mehra, M. P. Meredith, T. Meunier, T. Miles, J. M. Morell, L. Mortier, S. Nicholson, J. O'Callaghan, D. O'Conchubhair, P. Oke, E. Pallàs-Sanz, M. Palmer, J. Park, L. Perivoliotis, P.-M. Poulain, R. Perry, B. Queste, L. Rainville, E. Rehm, M. Roughan, N. Rome, T. Ross, S. Ruiz, G. Saba, A. Schaeffer, M. Schönau, K. Schroeder, Y. Shimizu, B. M. Sloyan, D. Smeed, D. Snowden, Y. Song, S. Swart, M. Tenreiro, A. Thompson, J. Tintore, R. E. Todd, C. Toro, H. Venables, T. Wagawa, S. Waterman, R. A. Watlington, and D. Wilson, "OceanGliders: A Component of the Integrated GOOS," *Frontiers in Marine Science*, vol. 6, p. 422, 2019.

[6] H. Stommel, "The Slocum Mission," *Oceanography*, vol. 2, no. 1, pp. 22–25, 1989.

[7] C. Whitt, J. Pearlman, B. Polagye, F. Caimi, F. Muller-Karger, A. Copping, H. Spence, S. Madhusudhana, W. Kirkwood, L. Grosjean, B. M. Fiaz, S. Singh, S. Singh, D. Manalang, A. S. Gupta, A. Maguer, J. J. H. Buck, A. Marouchos, M. A. Atmanand, R. Venkatesan, V. Narayanaswamy, P. Testor, E. Douglas, S. De Halleux, and S. J. Khalsa, "Future Vision for Autonomous Ocean Observations," *Frontiers in Marine Science*, vol. 7, p. 697, 2020.

[8] D. C. Webb, P. J. Simonetti, and C. P. Jones, "Slocum: An underwater glider propelled by environmental energy," *IEEE Journal of Oceanic Engineering*, vol. 26, no. 4, pp. 447–452, 2001.

[9] Teledyne Marine, "Slocum glider," https://www.teledynemarine.com/brands/webb-research/slocum-glider, 2025, accessed: 2025-12-01.

[10] T. Inanc, S. Shadden, and J. Marsden, "Optimal trajectory generation in ocean flows," in *American Control Conference (ACC)*, 2005.

[11] W. Zhang, T. Inanc, S. Ober-Blobaum, and J. E. Marsden, "Optimal trajectory generation for a glider in time-varying 2D ocean flows B-spline model," in *International Conference on Robotics and Automation (ICRA)*, 2008.

[12] D. Kruger, R. Stolkin, A. Blum, and J. Briganti, "Optimal AUV path planning for extended missions in complex, fast-flowing estuarine environments," in *International Conference on Robotics and Automation (ICRA)*, 2007.

[13] T. Lolla, M. P. Ueckermann, K. Yigit, P. J. Haley, and P. F. J. Lermusiaux, "Path planning in time dependent flow fields using level set methods," in *International Conference on Robotics and Automation (ICRA)*, 2012.

[14] M. Aguiar, J. B. de Sousa, J. M. Dias, J. E. da Silva, R. Mendes, and A. S. Ribeiro, "Optimizing autonomous underwater vehicle routes with the aid of high resolution ocean models," in *OCEANS*, 2019.

[15] E. Fernández-Perdomo, J. Cabrera-Gámez, D. Hernández-Sosa, J. Isern-González, A. C. Domínguez-Brito, A. Redondo, J. Coca, A. G. Ramos, E. Á. Fanjul, and M. García, "Path planning for gliders using regional ocean models: Application of Pinzón path planner with the

ESEOAT model and the RU27 trans-Atlantic flight data," in *OCEANS*, 2010.

[16] W. H. Al-Sabban, L. F. Gonzalez, and R. N. Smith, "Extending persistent monitoring by combining ocean models and Markov decision processes," in *OCEANS*, 2012.

[17] Y. Zhou, Y. Li, R. Juan, T. Wang, Z. Li, S. Liu, W. Ma, and Z. Gao, "Reinforcement learning-based path planning for underwater gliders using ocean current forecasts," *Ocean Engineering*, vol. 353, p. 124736, Apr. 2026.

[18] F. Lekien, L. Mortier, and P. Testor, "Glider Coordinated Control and Lagrangian Coherent Structures," *IFAC Proceedings Volumes*, vol. 41, no. 1, pp. 125–130, 2008.

[19] N. E. Leonard, D. A. Paley, F. Lekien, R. Sepulchre, D. M. Fratantoni, and R. E. Davis, "Collective Motion, Sensor Networks, and Ocean Sampling," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 48–74, 2007.

[20] N. E. Leonard, D. A. Paley, R. E. Davis, D. M. Fratantoni, F. Lekien, and F. Zhang, "Coordinated control of an underwater glider fleet in an adaptive ocean sampling field experiment in Monterey Bay," *Journal of Field Robotics*, vol. 27, no. 6, pp. 718–740, 2010.

[21] G. Han, Z. Zhou, T. Zhang, H. Wang, L. Liu, Y. Peng, and M. Guizani, "Ant-Colony-Based Complete-Coverage Path-Planning Algorithm for Underwater Gliders in Ocean Areas With Thermoclines," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 8, pp. 8959–8971, 2020.

[22] Mausam and A. Kolobov, *Planning with Markov Decision Processes: an AI Perspective*. Morgan & Claypool Publishers, 2012, vol. 17.

[23] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. MIT Press, 2018.

[24] J. A. Graham, E. O'Dea, J. Holt, J. Polton, H. T. Hewitt, R. Furner, K. Guihou, A. Brereton, A. Arnold, S. Wakelin, J. M. Castillo Sanchez, and C. G. Mayorga Adame, "AMM15: a new high-resolution NEMO configuration for operational simulation of the European north-west shelf," *Geoscientific Model Development*, vol. 11, no. 2, pp. 681–696, 2018.

[25] Met Office, "Met office marine data service," https://www.metoffice.gov.uk/services/data/met-office-marine-data-service, 2025, accessed: 2025-11-05.

[26] C. B. Browne, E. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton, "A survey of Monte Carlo tree search methods," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 4, no. 1, pp. 1–43, 2012.

[27] L. Merckelbach, "glidersim," https://github.com/smerckel/glidersim, 2025, accessed: 2025-11-05.

[28] ——, "Depth-averaged instantaneous currents in a tidally dominated shelf sea from glider observations," *Biogeosciences*, vol. 13, no. 24, pp. 6637–6649, 2016.

[29] L. Merckelbach, A. Berger, G. Krahmann, M. Dengler, and J. R. Carpenter, "A Dynamic Flight Model for Slocum Gliders and Implications for Turbulence Microstructure Measurements," *Journal of Atmospheric and Oceanic Technology*, vol. 36, no. 2, pp. 281–296, 2019.

[30] B. Wang, J. Xiong, S. Wang, D. Ma, and C. Liu, "Steady motion of underwater gliders and stability analysis," *Nonlinear Dynamics*, vol. 107, no. 1, pp. 515–531, 2022.

[31] H. Wu, J. Ding, W. Niu, M. Liao, Y. Song, L. Tan, and S. Yan, "Combination optimization method for motion trajectory of morphing underwater gliders balancing energy efficiency and motion accuracy," *Ocean Engineering*, vol. 329, p. 121154, 2025.

[32] M. Balandat, B. Karrer, D. Jiang, S. Daulton, B. Letham, A. G. Wilson, and E. Bakshy, "BoTorch: A framework for efficient Monte-Carlo Bayesian optimization," in *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[33] M. Olson, E. Santorella, L. C. Tiao, S. Cakmak, M. Garrard, S. Daulton, Z. J. Lin, S. Ament, B. Beckerman, E. Onofrey, P. Igusti, C. Lara, B. Letham, C. Cardoso, S. S. Shen, A. C. Lin, M. Grange, E. Kashtelyan, D. Eriksson, M. Balandat, and E. Bakshy, "Ax: a platform for adaptive experimentation," in *AutoML 2025 ABCD Track*, 2025.

[34] L. Kocsis and C. Szepesvári, "Bandit based Monte-Carlo planning," in *European Conference on Machine Learning (ECML)*, 2006.

[35] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time Analysis of the Multiarmed Bandit Problem," *Machine Learning*, vol. 47, no. 2, pp. 235–256, 2002.

[36] G. M. J.-B. Chaslot, M. H. M. Winands, and B. Bouzy, "Progressive strategies for Monte-Carlo Tree Search," *New Mathematics and Natural Computation*, vol. 4, no. 03, pp. 343–357, 2008.

[37] A. Couëtoux, J.-B. Hoock, N. Sokolovska, O. Teytaud, and N. Bonnard, "Continuous Upper Confidence Trees," in *Learning and Intelligent Optimization*, 2011, vol. 6683, pp. 433–445.

[38] T. Cazenave and N. Jouandeau, "On the Parallelization of UCT," in *Computer Games Workshop*, 2007.

[39] G. M. J.-B. Chaslot, M. H. M. Winands, and H. J. Van Den Herik, "Parallel Monte-Carlo Tree Search," in *Computers and Games*, 2008, vol. 5131, pp. 60–71.

[40] C. A. Harris, A. Lorenzo-Lopez, O. Jones, J. J. H. Buck, A. Kokkinaki, S. Loch, T. Gardner, and A. B. Phillips, "Oceanids C2: An Integrated Command, Control, and Data Infrastructure for the Over-the-Horizon Operation of Marine Autonomous Systems," *Frontiers in Marine Science*, vol. 7, p. 397, 2020.

[41] National Oceanography Centre Marine Autonomous and Robotic Systems Team, "Communications backbone," https://gitlab.com/nocacuk/mars-oss/cb/communications-backbone, 2026, accessed: 2026-01-06.

[42] ——, "Backbone message formats," https://gitlab.com/nocacuk/mars-oss/cb/backbone-message-format, 2026, accessed: 2026-01-06.

[43] National Oceanography Centre, "British Oceanographic Data Centre (BODC)," https://www.bodc.ac.uk/, 2025, accessed: 2025-11-05.

[44] Copernicus Authors, "Copernicus Monitoring Environment Marine Service (CMEMS)," https://marine.copernicus.eu/, 2025, accessed: 2025-11-05.

[45] C. C. Eriksen, T. J. Osse, R. D. Light, T. Wen, T. W. Lehman, P. L. Sabin, J. W. Ballard, and A. M. Chiodi, "Seaglider: A long-range autonomous underwater vehicle for oceanographic research," *IEEE Journal of Oceanic Engineering*, vol. 26, no. 4, pp. 424–436, 2001.

[46] D. M. Roijers, P. Vamplew, S. Whiteson, and R. Dazeley, "A Survey of Multi-Objective Sequential Decision-Making," *Journal of Artificial Intelligence Research*, vol. 48, pp. 67–113, 2013.

[47] Z. Sunberg and M. Kochenderfer, "Online algorithms for pomdps with continuous state, action, and observation spaces," in *International Conference on Automated Planning and Scheduling (ICAPS)*, 2018.

[48] D. Hafner, T. Lillicrap, I. Fischer, R. Villegas, D. Ha, H. Lee, and J. Davidson, "Learning latent dynamics for planning from pixels," in *International Conference on Machine Learning (ICML)*, 2019.

[49] J. Schrittwieser, I. Antonoglou, T. Hubert, K. Simonyan, L. Sifre, S. Schmitt, A. Guez, E. Lockhart, D. Hassabis, T. Graepel, T. Lillicrap, and D. Silver, "Mastering Atari, Go, chess and shogi by planning with a learned model," *Nature*, vol. 588, no. 7839, pp. 604–609, 2020.

[50] J. Xiao, V.-A. Darvariu, B. Lacerda, and N. Hawes, "Gaussian process aggregation for root-parallel Monte Carlo tree search with continuous actions," *arXiv preprint arXiv:2512.09727*, 2025.

[51] M. Painter, M. Baioumy, N. Hawes, and B. Lacerda, "Monte Carlo Tree Search with Boltzmann exploration," *Advances in Neural Information Processing Systems*, vol. 36, 2023.

[52] M. Budd, B. Lacerda, and N. Hawes, "Stop! planner time: Metareasoning for probabilistic planning using learned performance profiles," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2024.

[53] Y. Zhang, C. Rueda, B. Kieft, J. P. Ryan, C. Wahl, T. C. O'Reilly, T. Maughan, and F. P. Chavez, "Autonomous tracking of an oceanic thermal front by a Wave Glider," *Journal of Field Robotics*, vol. 36, no. 5, pp. 940–954, 2019.

[54] J. Fonseca, M. Aguiar, J. B. D. Sousa, and K. H. Johansson, "Algal Bloom Front Tracking Using an Unmanned Surface Vehicle: Numerical Experiments Based on Baltic Sea Data," in *OCEANS*, 2021.

[55] R. D. Patmore, D. Ferreira, D. P. Marshall, M. D. du Plessis, J. A. Brearley, and S. Swart, "Evaluating existing ocean glider sampling strategies for submesoscale dynamics," *Journal of Atmospheric and Oceanic Technology*, vol. 41, no. 7, pp. 647–663, 2024.