EA-based ASV Trajectory Planner for Detecting Cyanobacterial Blooms in Freshwater

Gonzalo Carazo-Barbero
gocarazo@ucm.es
Universidad Complutense de Madrid
Madrid, Madrid, Spain

Eva Besada-Portas
ebesada@ucm.es
Universidad Complutense de Madrid
Madrid, Madrid, Spain

José Luis Risco-Martín
jlrisco@ucm.es
Universidad Complutense de Madrid
Madrid, Madrid, Spain

José Antonio López-Orozco
jalo@ucm.es
Universidad Complutense de Madrid
Madrid, Madrid, Spain

ABSTRACT

Cyanobacterial Blooms (CBs) constitute a relevant ecological and public health problem since they often produce toxic metabolites that endanger the lives of many species, and they prevent human water consumption and recreational use. To determine the locations of CBs in lentic water bodies, we present a new planner based on Evolutionary Algorithms (EAs) that optimizes the trajectory of an Autonomous Surface Vehicle (ASV) equipped with a probe capable of detecting CBs. The planner 1) exploits the information provided by a particle transport simulator that determines the CB distribution from the water currents and the inherent CB behavior (in particular, its biological growth and vertical displacements) and 2) is supported by an EA that optimizes the mission duration, the ASV trajectory length, and the contributions of each simulated particle to the predicted cyanobacterial concentration along the ASV trajectory. The planner also ensures the trajectory feasibility from the ASV, probe, and water body perspective; and refines the trajectory shape by increasing the number of the decision variables during the iteration of an EA supported by usual NSGA-II operations. The results over different scenarios show that the planner determines overall good solutions that adapt the ASV trajectory to the evolution of CB distribution.

CCS CONCEPTS

• **Computing methodologies** → *Continuous models*; **Motion path planning**; **Optimization algorithms**.

KEYWORDS

Genetic Algorithms, Multi-objective Optimization, Decision Making, Earth Sciences and the Environment, Robotics

GECCO '23, July 15–19, 2023, Lisbon, Portugal

ACM ISBN 979-8-4007-0119-1/23/07.
https://doi.org/10.1145/3583131.3590484

ACM Reference Format:
Gonzalo Carazo-Barbero, Eva Besada-Portas, José Luis Risco-Martín, and José Antonio López-Orozco. 2023. EA-based ASV Trajectory Planner for Detecting Cyanobacterial Blooms in Freshwater. In *Genetic and Evolutionary Computation Conference (GECCO '23), July 15–19, 2023, Lisbon, Portugal.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3583131.3590484

1 INTRODUCTION

Fresh water is a fragile resource that must be frequently monitored to guarantee that its quality is appropriate for human consumption and recreational uses. The scarcity of good-quality inland water bodies is due to many factors, such as misuse, pollution, and extreme floods and droughts caused by climate change. The importance of the problem, which also affects the life of the species around or within the water body, is remarked by the sixth goal of the United Nations 2030 Agenda for Sustainable Development [31].

Cyanobacterial Blooms (CBs) are a relevant ecological and public health problem [20] because they can produce secondary toxic metabolites, which put in risk the lives of many species, and the drinking and recreational use of water. They can cause more problems, such as their accumulation in the treatment plants or the reduction of the oxygen dissolved in the water. Hence, the new European Directive on the Quality of Water for Human Consumption, which came into force on January 2021, recommends adequately monitoring freshwater bodies at risk of CBs [12]. To follow this recommendation, the institutions responsible for Water, Health, and Environment must use new tools that, with reasonable costs, facilitate the early prediction and reliable detection of CBs.

However, it is not trivial to predict a CB's location due to the phenomenon's complexity and the necessity to have sufficient data to know, with adequate granularity, the water body conditions [28]. This happens because 1) CBs involve biological processes dependent on many factors (e.g., climate and meteorology, water body morphology, and nutrients) and 2) CBs' location/dispersion is affected by water currents and their ability to move vertically (for photosynthesizing near the water surface and capturing nutrients at greater depths). Besides, traditional monitoring methods (e.g., taking water samples manually or automatically measuring water parameters from the fixed stations of early warning networks) do not provide enough information to determine when and where the conditions that cause the CB occur.

An alternative to facilitate the detection of CBs is to ask an Autonomous Surface Vehicle (ASV, a type of robotized boat) equipped

with adequate probes to move to the "best" locations to take the measurements. To achieve it, the ASV has to be equipped with a self-driving system that incorporates advanced location, perception, planning, guidance, navigation, and control techniques, especially intended for each type of ASV mission [18]. In this context, this paper presents a new planning technique for optimizing, before the mission starts, the trajectory of an ASV intended for CB detection.

To develop this planner, the CB detection with ASVs is defined as a multi-objective constrained optimization problem that is solved with an Evolutionary Algorithm (EA). Previous works have followed other bio-inspired approaches, usually focused on different types of ASV water monitoring missions. For instance, when no information about the pollutant distribution is available, the problem has been set up as a coverage problem and solved with Genetic Algorithms (GA, [2, 3]). Otherwise, when the information to be gathered in the regions of interest is relevant, it has been set up as an information gain problem and solved with Ant Colony Optimization (ACO, [33]) and Particle Swarm Optimization (PSO, [34]). Besides, PSO and GA have been used to determine the trajectories of multiple ASVs according to the water quality uncertainty modeled with a Gaussian Process and updated with the new ASVs measurements [16, 17, 22]. Finally, ACO has also been used to define the best visiting ordering of the water monitoring points provided by an operator [32]. Our EA-based planner differs from the previous approaches, as it considers the CB concentration and distribution obtained by a biological and physical simulator to simultaneously optimize 1) the cyanobacterial concentration expected to be observed with the ASV probes, 2) the ASV trajectory length, and 3) the mission duration. Moreover, the planner is supported by a variant of NSGA-II [11] that sequentially increments the number of decision variables that define the ASV trajectory to refine its shape and accelerate/improve the results of the planner.

Besides, our approach is inspired by the EA-based planner for *generic* pollutant detection presented in [8], diverging from it by incorporating in the simulator/EA specific behaviors/objectives of/for CBs, exploiting the multi-objective and constraint handling mechanisms of NSGA-II, and including the trajectory refinement process. Finally, and from a broader perspective, trajectory planners for water monitoring with autonomous underwater vehicles are reviewed in [21], EA-based approaches for applications related to water resources are analyzed in [15], and NSGA-II is commonly used for missions involving autonomous (or unmanned) vehicles [13, 23–25, 29]. Our work is closer to this last group, although as the type of mission, objective functions, constraint indicators and trajectory encoding are different is not comparable to them.

2 PROBLEM DESCRIPTION

This section presents the main elements and variables of the problem, describes the models used to simulate the CB behavior, and introduces the CB simulations used in this paper to test the planner.

2.1 Problem Elements and Variables

To efficiently determine when and where CBs are happening, our approach sends a single ASV equipped with a probe of adjustable depth to observe those locations where the simulation predicts that these phenomena will occur.

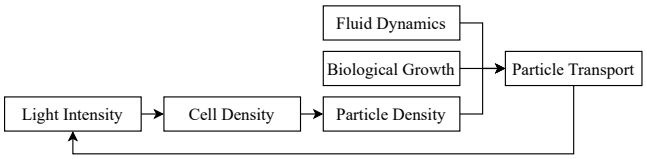


Figure 1: Relationships among the models

In other words, it determines the 3D trajectory of the probe ($\boldsymbol{s}(t) = [s_x(t), s_y(t), s_z(t)]$), defined by the coordinates of the ASV at the water body surface ($s_x(t), s_y(t)$) and the depth of the probe ($s_z(t)$), taking into account the spatiotemporal distribution of the CB concentration ($c(t, x, y, z)$). To calculate this distribution, it computes the 3D locations ($\boldsymbol{p}^k(t) = [p_x^k(t), p_y^k(t), p_z^k(t)]$ with $k \in [1, ..., N_p(t)]$) of a set of $N_p(t)$ CB particles, taking into account their initial locations, their biological growth rate (Δ_p), the effect of the solar radiation at different depths ($I(t, z)$) in the density of the particles ($\rho_p^k(t)$), and the steady state velocity of the water flow $\boldsymbol{u}_{ss}(\boldsymbol{w}) = [u_{ss,x}(\boldsymbol{w}), u_{ss,y}(\boldsymbol{w}), u_{ss,z}(\boldsymbol{w})]$ at any location $\boldsymbol{w} = [x, y, z]$ of the water body. Finally, it is worth noting that the initial starting and ending points ($\boldsymbol{s}(t_0)$ and $\boldsymbol{s}(t_e)$) of the ASV trajectory are fixed, and that the planner wants to make the mission duration ($t_e - t_0$) and ASV trajectory as short as possible, as far as it can observe areas of high cyanobacterial concentration $c(t, x, y, z)$.

Besides, and for notation purposes, $\dot{\boldsymbol{s}}(t)$ and $\ddot{\boldsymbol{s}}(t)$ stand for the ASV and probe velocity and accelerations, and $\boldsymbol{u}_p^k(t) = \dot{\boldsymbol{p}}^k(t)$ for the velocities of the cyanobacteria particles. Finally, in some cases the variable's dependencies will be dropped to shorten the expressions.

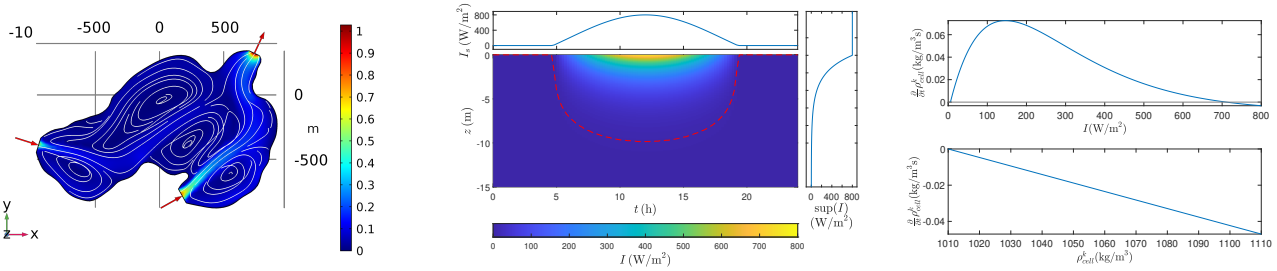
2.2 Models and Simulation of the CB

In order to simulate the evolution of the distribution of the CB, we combine/feedforward the results obtained by the group of biological and physical models which are represented, in the order they have to be invoked, in Fig. 1. In particular, to simulate the CB displacement, we assume that the bloom is formed by a set of particles whose locations are affected by the water-body domain, the fluid dynamics, and CB flotability, this last dependent on the cells density and on the luminosity reaching them. Besides, to simulate its biological growth, we add or remove particles probabilistically, according to a constant growth rate given by a Gaussian distribution. All these models, except the fluid dynamics, are implemented in Matlab [19].

2.2.1 The Physical Domain. It is given as a Computer-Aided Design (CAD) model with multiple boundaries. For the examples in this paper, we use a 3D domain that has a surface area (represented in Fig. 2a) of 1.43×10^6 m^2 , a volume of 6.61×10^7 m^3, a maximum depth of 70.79 m, a top boundary for the water surface, a bottom boundary for its bed, and side boundaries for its inlets and outlets.

2.2.2 Fluid Dynamics. They are modeled as in [8] using the Navier-Stokes equations for incompressible fluids [6] and simulated using COMSOL Multiphysics [9]. In particular, the fluid dynamics of the water body in this paper are simulated, creating within COMSOL a laminar and steady-state flow[1] model, importing the CAD file of the water domain, and setting the fluid material to water. Next, a

[1]This is reasonable for small time frames and slow moving water.



(a) Solution of the fluid flow

(b) Light intensity over one day

(c) Variations on cell density

Figure 2: Stationary fluid flow solution, light intensity during one day at varying depths, and variations on cell density

laminar inflow rate of 330 m^3/s is imposed on the left-most inlet and 600 m^3/s on the other, sliding conditions on the water surface, non-sliding on the bed, and laminar flow at null pressure on the outlet. Finally, the model is sequentially refined and solved over three meshes of increasing resolution. In the end, we obtain the $\boldsymbol{u}_{ss}(\boldsymbol{w})$ schematized in Fig. 2a, using a blue-red color scale to depict the flow velocity magnitude (in m/s) on the water body surface and white stripes to show the streamlines, which correspond to 1) two currents coming from each inlet and converging before heading to the outlet (all marked in red); and 2) several slow-moving vortices that can trap the particles instead of letting them exit the domain.

2.2.3 Density of the Cyanobacteria Colony (CC). We follow the process in [1], outlined next, to calculate the density ρ_p^k of each CC.

First, Beer-Lambert law [7] with a light attenuation coefficient of 0.5 m^{-1} is used to determine the light intensity $I(z,t)$ at depth z of the CC and time t of the day[2]. This is represented in Fig. 2b, where the top plot shows the light irradiance at the surface along a typical day, the middle colored map the light intensity reaching each depth along the day, the red dashed line in the colored map the luminosity threshold I_{lim} (discussed next) for light/dark conditions, and the right plot the maximal light intensity at different depths.

Next, the instant rate of change of the density of the cells ρ^k_{cell} that form the CC is calculated using the Euler method and $I(z,t)$, which is related to the accumulation or release of carbohydrates and which falls in one of two regimes. On light conditions ($I(z,t) > I_{lim}$), the variation of ρ^k_{cell} is ruled by $I(z,t)$ and generally increases the cell's density (although it can also decrease it if the intensity is too high). On dark conditions ($I(z,t) < I_{lim}$), the change of ρ^k_{cell} depends on ρ^k_{cell} itself, decreasing faster at greater densities and not going below a minimum value. Equation (1) models these behaviors,

$$\begin{cases} \frac{\partial}{\partial t}\rho^k_{cell}(t) = \beta I^k \mathrm{e}^{-\frac{I^k}{I_0}} + \gamma & \text{if} \quad I(z,t) \geq I_{lim} \\ \frac{\partial}{\partial t}\rho^k_{cell}(t) = -\alpha\Delta_\rho \mathrm{H}\left(\Delta_\rho\right) & \text{if} \quad I(z,t) < I_{lim} \end{cases} \tag{1}$$

where α is the decay rate of cell density, β the normative factor in cell density, γ the mean rate of change in dark conditions, I_0 the photo-inhibition limit, $I^k = I(p^k_z(t), t)$ the light intensity that reaches the particle, $\Delta_\rho = \rho^k_{cell} - C_p$, C_p is the minimum cell density, and H$(\cdot)$ the Heaviside function. Figure 2c represents the variations in ρ^k_{cell} on light and dark conditions, which are respectively shown against I and ρ^k_{cell} on the top and bottom plots.

Finally, we obtain the CC density ρ_p^k with Eq. (2), combining the density of the cells with the density of the mucilage (a gelatinous substance that surrounds the cells) and that of the air bubbles formed by the oxygen released by the cyanobacteria,

$$\rho_p^k(t) = \rho^k_{cell}(t) \cdot n_{cell}(1 - n_{gas}) + \rho_{muc}(1 - n_{cell}), \tag{2}$$

where n_{cell} and n_{gas} are the proportion of the colony formed by cells and gas vacuoles respectively, ρ_{muc} the density of the mucilage, and $k \in [1, ..., N_p(t)]$ the particle index.

2.2.4 Biological Growth of the CC. We use a growth model where the rate of change of the number of particles Δ_p remains constant[3]. In particular, on each simulation time-step and for each particle, a random number r^k_Δ is taken from a Gaussian distribution $\mathcal{N}(\mu_\Delta, \sigma_\Delta)$ of mean μ_Δ (equal to the expected growth per time-step) and spread σ_Δ (inversely proportional to the time required to renew the population of particles completely). Next, the number of particles to be created or destroyed for each particle is obtained as

$$N^k_{new} = \mathrm{trunc}\left(r^k_\Delta\right) + \mathrm{sign}\left(r^k_\Delta\right) H\left(s^k_\Delta < \mathrm{mod}\left(|r^k_\Delta|, 1\right)\right), \tag{3}$$

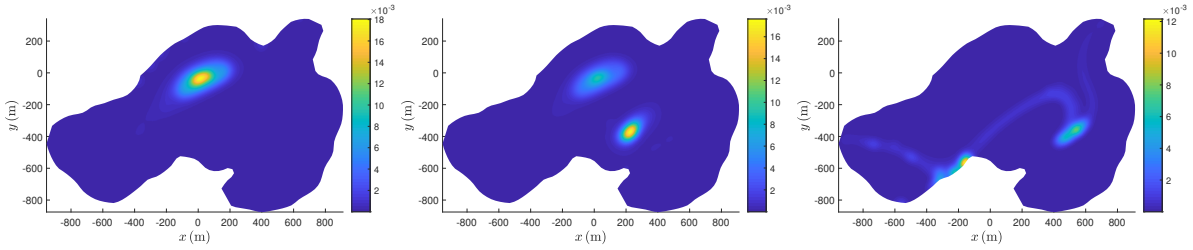
where $s^k_\Delta \in [0,1]$ is sampled from the uniform distribution $\mathcal{U}([0,1])$, $H(\cdot) = 1$ if the inequality holds and $H(\cdot) = 0$ otherwise, and trunc$(\cdot)$ is the truncation function that rounds the input towards zero. When N^k_{new} is a positive integer, new particles are created around the parent, keeping the parent's velocity and density; when N^k_{new} is a negative integer, the closest particles to the parent are deleted; and when $N^k_{new} = 0$ the number of particles remains unchanged.

2.2.5 Cyanobacterial Transport. After obtaining $\boldsymbol{u}_{ss}(\boldsymbol{w})$ and $\rho_p^k(t)$, and applying the biological growth, the CC transport is simulated with Eq. (4), which is similar to the force balance expression of [8], except for the use of an additional random walk diffusion term and of a changing $\rho_p^k(t)$ that induces the CC auto-vertical displacement,

$$\left(1 + \frac{1}{2}\frac{\rho_f}{\rho_p^k}\right)\frac{d\boldsymbol{u}_p^k}{dt} = \left(1 + \frac{\rho_f}{\rho_p^k}\right)\boldsymbol{g} + \frac{18\nu\rho_f}{d_p^2\rho_p^k}\left(\boldsymbol{u}_{ss} - \boldsymbol{u}_p^k\right) + \frac{3}{2}\frac{\rho_f}{\rho_p^k}\frac{D\boldsymbol{u}_{ss}}{Dt} + D_0 r_D, \tag{4}$$

[2] Since the light reaching the water surface depends on the time of day, season and weather conditions.

[3] This differs from other models such as [26], which consider that Δ_p depends on the nutrient concentration and the water temperature.



(a) Simulation A (b) Simulation B (c) Simulation C

Figure 3: Physical simulations

where ρ_f is the water density, $\boldsymbol{g}$ the gravitational acceleration vector, ν the kinematic viscosity of the fluid, d_p the particle diameter, operator $\frac{d}{dt} = \frac{\partial}{\partial t} + \left(\boldsymbol{u}_p^k \cdot \nabla\right)$ the time derivative along the trajectory of a particle, $\frac{D}{Dt} = \frac{\partial}{\partial t} + (\boldsymbol{u}_{ss} \cdot \nabla)$ the convective derivative along the path of the fluid, D_0 the diffusion coefficient, and r_D a random 3D vector taken from a Gaussian distribution of mean [0,0,0] and standard deviation [1,1,1].

To simulate this model, we consider that there are $N_p(t)$ CC particles respectively placed at $\boldsymbol{p}^k(t)$ and use the explicit Euler method to integrate the acceleration of each particle $\frac{d\boldsymbol{u}_p^k}{dt}$, obtained with Eq. (4) from the previous time step's velocity $\boldsymbol{u}_p^k(t)$ and CC density ρ_p^k, and steady-state fluid velocity $\boldsymbol{u}_{ss}$. This operation returns the new particle velocity $\boldsymbol{u}_p^k(t+\Delta t)$, which is integrated again to obtain the new position $\boldsymbol{p}^k(t+\Delta t)$ of the particle[4].

2.2.6 Concentration of CC. In this stage, a Gaussian function is assigned to each particle with a standard deviation $\boldsymbol{\sigma_c} = [\sigma_{c,x}, \sigma_{c,y}, \sigma_{c,z}]$ and an amplitude $A_c = \frac{1}{(2\pi)^{3/2}\sigma_{c,x}\sigma_{c,y}\sigma_{c,z}}$ such that the integral of each Gaussian is equal to one and represents the probability of detecting each particle. Next, and according to Eq. (5), we calculate the concentration of CCs at a given point and time by adding up the Gaussian functions of each particle.

$$c(x,y,z,t) = \sum_{k=1}^{N_p(t)} A_c \exp\left(\sum_{d\in\{x,y,z\}} -\frac{\left(d - p_d^k(t)\right)^2}{2\sigma_{c,d}^2}\right) \quad (5)$$

2.3 CB Evolution in the Scenarios under Study

Using the fluid flow summarized in Fig. 2a, we set up and simulate three different scenarios to obtain the CC concentrations $c(x,y,z,t)$ that will be later used to test the planer in Section 4. In the following, we first detail the particularities of each scenario and afterwards we present their common characteristics.

Simulation A. It contains 500 initial particles, whose initial positions $\boldsymbol{p}^k(t_0)$ are sampled according to a Gaussian distribution[5] centered at the flow vortex placed at $[0, 0, -2]$ m and with a standard deviation of $[25, 25, 0.5]$ m.

[4]We also consider that our water body surface boundary supports the horizontal sliding and sinking of the particles (to prevent them from coming out of the water), that the outlet boundary is transparent and lets the particles exit the domain; and that the remaining boundaries are non-sliding boundaries (forcing particles that reach them to remain immobile unless they separate from them).

[5]Particles outside the water body are re-sampled to make them start at valid locations.

Table 1: Fixed parameters of the simulations

Parameter	Value	Parameter	Value
Δt	60 s	n_{gas}	5 %
α	4.7×10^{-4} s^{-1}	ρ_{muc}	998 kg/m^3
β	1.5×10^{-3} s^2/m^3	ρ_f	1000 kg/m^3
γ	-8.3×10^{-3} kg/m^{3}s	ν	2.5×10^{-4} m$^2/s$
C_p	1010 kg/m^3	d_p	8×10^{-4} m
I_{lim}	5.75 W/m^2	D_0	1 m/s^2
I_0	146.43 W/m^2	$\boldsymbol{\sigma_c}$	[25, 25, 0.5] m
n_{cell}	10 %		

Simulation B. It uses the same distribution as in Simulation A to sample 250 particles, and another Gaussian distribution centered at the second vortex (placed at $[230, -365, -2]$ m) with a standard deviation of $[25, 25, 1]$ for the remaining 250 particles.

Simulation C. It contains 500 particles sampled from a Gaussian distribution centered at the inlet of the lake placed at $[-930, -400, -1]$ m with a standard deviation of $[25, 25, 0.5]$ m. This situation makes the particles follow the rapidly-moving main current of the lake.

In the three scenarios, the initial ρ_{cell}^k are set such that the particles are neutrally buoyant after computing the CC density using Eq. (2). The rest of the parameters used for these simulations are displayed in Table 1. Finally, Fig. 3 shows with a color scale, the CC concentration integrated over the water column and averaged over a period of three days in Simulations A and B, and one day in Simulation C. The yellow areas represent columns of high concentration, while the blue areas show columns of low or null concentration.

3 EA FOR PLANNING ASV TRAJECTORIES

This section presents the main features of our planner. With that purpose, it briefly describes the encoding of the 3D trajectories (directly manipulated by the EA) of the ASV and the probe; introduces the objective and constraint functions used to evaluate them; and presents the main steps of its supporting EA, which optimizes the 3D trajectories while successively refining its encoding.

3.1 Trajectory Encoding

The 3D trajectories $\boldsymbol{s}(t)$ optimized by EA-based planners (which, in our case, define the ASV surface location and probe depth) are often codified with splines curves, as they help EAs to determine

continuous smooth trajectories easy to follow by a vehicle [27]. Besides, the range of characteristics of the curves of this family allows for selecting the best type for each problem.

We use a 3D interpolating cubic cardinal spline [30], with a zero tension parameter and N_f segments, which is built by fixing the position of the first and last nodes (i.e., of the start and ending points of ASV trajectory $\boldsymbol{s}(t)$) and by fixing to zero the initial and final velocities (to have a trajectory that starts and ends with the ASV at rest). We also let the planner set the position and timings of all intermediate nodes and the last node's temporal parameter to optimize the trajectory and mission duration. Since each node is defined by its x, y, z, and t coordinates, and, of those, the first node is completely fixed, and the last node only accepts variations in time, this type of spline can be codified by $4\cdot(N_f-1)+1$ continuous variables. Finally, although $\boldsymbol{s}(t)$ is a continuous function over time, it is discretized at periodic time steps (i.e at $t = t_j = t_0 + j \cdot T_s$, with $T_s = \sup(\frac{t_e-t_0}{10^4}, \frac{t_e-t_0}{L})$ and L the length of $\boldsymbol{s}(t)$), in order to evaluate some criteria of the EA.

3.2 Evaluation Criteria

This section describes the Constraint and Optimization Functions (CFs and OFs) used by the EA to grade $\boldsymbol{s}(t)$ and determine which of the solutions of the population should survive and be manipulated to define a 3D probe & ASV trajectory that efficiently explores the water body. Note that to gain space for presenting the novelties of this work we explain in greater detail those functions specially developed in this planner for evaluating $\boldsymbol{s}(t)$ according to its capability to facilitate the detection of CBs. Finally, also note that the CFs, as stated in this paper, return their violation degree (i.e., how far off they are from being fulfilled).

3.2.1 Maximal Mission Duration CF. It is used to check if the ASV is performing the mission in the allowed time $T_{mission}$.

$$\mathrm{CF}_1 = \sup(0, t_e - t_0 - T_{mission}) \tag{6}$$

3.2.2 Maximal ASV Velocity CF. It checks if the horizontal ASV velocity (obtained by evaluating the derivative of the spline at $t_j \in [t_0 : T_s : t_e]$) relative to the fluid flow is lower than the maximal allowed ASV speed $v_{max,H}$,

$$\mathrm{CF}_2 = \sup\left(0, \sup_{t_j\in[t_0:T_s:t_e]} \left(\left\|\dot{\boldsymbol{s}}_H(t_j) - u_{ss,H}(t_j)\right\|\right) - v_{max,H}\right), \tag{7}$$

where $\boldsymbol{s}_H(t) = \left[s_x(t), s_y(t)\right]$, $\boldsymbol{u}_{ss,H} = \left[u_{ss,x}(\boldsymbol{w}_s(t)), u_{ss,y}(\boldsymbol{w}_s(t))\right]$, and $\boldsymbol{w}_s(t) = \left[s_x(t), s_y(t), 0\right]$.

3.2.3 Maximal ASV Acceleration CF. It checks if the horizontal ASV acceleration with respect to the fluid flow is held within the symmetric interval $[-a_{max,H}, a_{max,H}]$.

$$\mathrm{CF}_3 = \sup\left(0, \sup_{t_j\in[t_0:T_s:t_e]} \left(\left\|\ddot{\boldsymbol{s}}_H(t_j) - \frac{D\boldsymbol{u}_{ss,H}(t_j)}{Dt}\right\|\right) - a_{max,H}\right) \tag{8}$$

3.2.4 Maximal ASV Curvature CF. It checks if the horizontal unsigned curvature of the ASV trajectory is kept under the allowed curvature κ_{max}.

$$\mathrm{CF}_4 = \sup\left(0, \sup_{t_j\in[t_0:T_s:t_e]} \left(\frac{\left|\dot{s}_x(t_j)\ddot{s}_y(t_j) - \dot{s}_y(t_j)\ddot{s}_x(t_j)\right|}{\left(\dot{s}_x^2(t_j) + \dot{s}_y^2(t_j)\right)^{3/2}}\right) - \kappa_{max}\right) \tag{9}$$

3.2.5 Maximal Probe Velocity CF. The probe vertical velocity is also constrained under $v_{max,V}$ to fulfill its mechanical requirements.

$$\mathrm{CF}_5 = \sup\left(0, \sup_{t_j\in[t_0:T_s:t_e]} \left(\left|\dot{s}_z(t_j)\right|\right) - v_{max,V}\right) \tag{10}$$

3.2.6 Maximal Probe Acceleration CF. The vertical acceleration of the probe is constrained too within the range $[-a_{max,V}, a_{max,V}]$[6].

$$\mathrm{CF}_6 = \sup\left(0, \sup_{t_j\in[t_0:T_s:t_e]} \left(\left|\ddot{s}_z(t_j)\right|\right) - a_{max,V}\right) \tag{11}$$

3.2.7 Spatial Domain CF. It checks if the probe is contained within the boundaries of the spatial domain, which is smaller than the water-body domain, for the following reasons. On the one hand, we limit the depth to the range $[-15, 0]$ m due to account for the probe mechanism's reach and to reduce the surface area of the water body to ensure that the probe can reach its final location without getting captured in the lake shores. On the other hand, our planner also supports non-navigable areas, defined as boolean functions that take an $[x, y]$ position as input and return a positive number if the area is navigable and a negative number otherwise. In particular, this CF accounts for the length of $\boldsymbol{s}(t)$ that is out of the allowed domain (which is defined between the t_{l-1} and t_l instants in which $\boldsymbol{s}(t)$ intersects[7] with the boundary limits).

$$\mathrm{CF}_7 = \sup\left(0, \left(\sum_{l=1:2:N_{inter}} \int_{t_{l-1}}^{t_l} \|\dot{\boldsymbol{s}}(t)\|\, dt\right)\right) \tag{12}$$

3.2.8 Minimizing the Mission Duration OF. It is included in the planner to make the ASV perform the mission as soon as possible.

$$\min \mathrm{OF}_1 = (t_e - t_0) \tag{13}$$

3.2.9 Minimize the Total Trajectory Length OF. It is used in the planner to avoid unnecessary ASV displacements.

$$\min \mathrm{OF}_2 = L = \left(\int_{t_0}^{t_e} \|\dot{\boldsymbol{s}}(t)\|\, dt\right) \tag{14}$$

3.2.10 Maximize the Maximal Contribution of each Particle to the CC Concentration along the trajectory OF. The aim of this OF is to guide the ASV and the probe to traverse regions with high CC concentration $c(x, y, z, t)$. We have observed that functions that directly rely on this variable, such as the mean or the integral of the concentration along the trajectory, tend to get over-focused on the point of highest concentration for the maximum allowed time and disregard other regions of interest that also have a high concentration. Hence, for exploration purposes, we relate this OF to the maximum contribution of each particle to the concentration at any point along the trajectory, since spending more time in the

[6] Hence CF_2, CF_3 and CF_4 are used to check if $s(t)$ is feasible from the point of view of the ASV maneuverability, and CF_5 and CF_6 if it is feasible for the probe mechanism.
[7] An approach to determine these intersections efficiently is already presented in [8].

same region provides only marginal improvements as long as there are still particles to be detected that can contribute to the OF if the ASV approaches to them.

$$\max \mathrm{OF}_3 = \sum_{k=1}^{N_p(t)} \sup_{t_j \in [t_0:T_s:t_e]} \left[A_c \exp\left(\sum_{d \in \{x,y,z\}} -\frac{\left(s_d(t_j) - p_d^k(t_j)\right)^2}{2\sigma_{c,d}^2} \right) \right] \tag{15}$$

3.3 Evolutionary Algorithm

The EA that determines the best control points of the 3D spline curve followed by the ASV & the probe to detect the CC is implemented in Matlab and schematized in Algorithm 1. From lines 2 to 13, it implements the usual steps of NSGA-II[8], and from lines 14 to 20, the additional steps that carry out the trajectories refinement.

3.3.1 Population generation. Each trajectory of the population is defined by sampling the values of the extreme times of $\boldsymbol{s}(t)$ and of its interior control points $\boldsymbol{c}^i$ and times t^i, from uniform distributions of different ranges: c_x^i and c_y^i are limited by the extremes of the water body, c_z^i by the $[-15, 0]$ m probe range, and t^i by $[0, T_{max}]$. We also sort and slide the whole time vector by decreasing all their values by $t_0 - t_0^{mission}$ to make the initial trajectory time t_0 equal to the starting mission time $t_0^{mission}$. This process initializes the first population and brings N_i new solutions (immigrants) to all the generations of the algorithm.

3.3.2 Solution Evaluation. This phase evaluates the parameters associated with each individual of the population, including the CFs, the OFs, the Pareto front index, and the crowding distance [11]. For the evaluation of the Pareto fronts, any individual that meets the constraints (i.e. that has all CF_i equal to zero) dominates any solution that does not. When none of them meets them, the Pareto domination is determined based on CFs, and when all of them do it, it is based on OFs.

3.3.3 Parent Selection. It creates N_c pairs of candidate solutions, half of them selected to become parents using binary tournament [14], by comparing (as NSGA-II [11] proposes) their Pareto front index and, when the index are equal, their crowding distance.

3.3.4 Crossover. First, according to a crossover probability p_{cross}, we decide if each pair of parents undergoes a mating step or becomes a pair of children directly. Next, mating pairs are crossed using a two-point crossover[9]. Afterward, we sort the time values of each child, and when a child has two equal time values, one of them is averaged with the previous.

3.3.5 Mutation. A few genes of the children are 1) randomly sampled from a uniform probability of mutation p_{mut} that is inversely proportional to the number of decision variables, and 2) incrementally modified according to the displacement sampled from a Gaussian distribution with zero mean and standard deviations of $c_x^i = c_y^i = 200$ m, $c_z^i = 2$ m, and 2000 s for the times. Afterward, the time vector is sorted and slid to equal its first value to $t_0^{mission}$.

3.3.6 Recombination. The old population (*pop*) and the one that includes the children and the immigrants (*new*) are merged and sorted according to NSGA-II recombination approach, maintaining fixed the size of the population N_s.

3.3.7 Stop Condition. The algorithm ends when the maximum number of generations N_g has been reached.

3.3.8 Refinement. This process adds a new node to the trajectory spline $\boldsymbol{s}(t)$, increasing by one the number of segments N_f. The new node is added at the trajectory point where the cyanobacterial concentration $c(x, y, z, t)$ is highest as long as no other node is closer than 150 s in the time coordinate. The N_g generations are divided evenly in $N_f^{end} - N_f^0 + 1$ periods and the refinement process is executed at the beginning of each period (except the first) such that each of the periods optimizes a level of refinement (splines with different N_f).

Algorithm 1: NSGA-II based planner

Input: *Dom*, Water and NNZ Domain
Input: $c(x, y, z, t)$, Estimated CC evolution
Input: $[N_f^0, N_f^{end}]$, Range of spline segments in $s(t)$
Input: N_s, Population size
Input: N_c, Number of children
Input: p_{xover}, Probability of crossover
Input: N_g, Number of generations
Result: *pop*, Population of solutions, sorted in Pareto fronts
Result: *val*, CF and OF values of pop, and their front index

1 $N_f \leftarrow N_f^0$
2 $pop \leftarrow GeneratePopulation(N_f, N_s, Dom)$
3 $val \leftarrow Evaluate(pop, c(x, y, z, t), Dom)$
4 $N_i \leftarrow N_s - N_c$ // Number of immigrants
5 $p_{mut} \leftarrow \frac{1}{(4(N_f-1)+1)}$
6 **for** $i \in [1, ..., N_g]$ **do**
7 $\quad parents \leftarrow ParentSelection(pop, val, N_c)$
8 $\quad children \leftarrow Crossover(parents, p_{xover})$
9 $\quad children \leftarrow Mutation(children, p_{mut})$
10 $\quad immigrans \leftarrow GeneratePopulation(N_f, N_i, Dom)$
11 $\quad newpop \leftarrow [children; immigrants]$
12 $\quad newval \leftarrow Evaluate(newpop, c(x, y, z, t), Dom)$
13 $\quad [pop, val] \leftarrow Recombine(pop, val, newpop, newval)$
14 $\quad$ **if** $RefineConditionsMet(pop, val)$ **then**
15 $\quad\quad N_f \leftarrow N_f + 1$
16 $\quad\quad p_{mut} \leftarrow \frac{1}{(4(N_f-1)+1)}$
17 $\quad\quad pop \leftarrow Refine(pop)$
18 $\quad\quad val \leftarrow Evaluate(pop, c(x, y, z, t), Dom)$
19 $\quad$ **end**
20 **end**

4 RESULTS

In this section we complete the characteristics of the three scenarios of the paper, show some illustrative results, and perform a comparative analysis of the algorithm with and without refinement.

[8] Although many EAs have been developed after NSGA-II [11], this technique is often and still used in trajectory planners due to their good results in different types of problems involving unmanned vehicles [13, 23–25, 29].
[9] The selectable cutting points make the information of the whole control node go to the same child.

4.1 Scenarios under Study

Three scenarios are presented in this paper, each one defined by a CC concentration $c(x, y, z, t)$, a pair of initial and final trajectory locations $s(t_0)$ and $s(t_{end})$, and a mission start time $t_0^{mission}$. The other mission parameters are shared: the maximal horizontal and vertical velocities $v_{max,H} = v_{max,V} = 1\,\text{m/s}$, the maximal horizontal and vertical accelerations $a_{max,H} = a_{max,V} = 0.1\,\text{m/s}^2$, the maximal curvature $\kappa_{max} = 0.2\,\text{m}^{-1}$, and the mission duration $T_{mission} = 3\,\text{h}$. Besides, in addition to the safety margins to the shore and the vertical probe range discussed in Section 3.2.7, a recreational area, on the right side of the lake, is set off limits.

Scenario A. It uses Simulation A, where concentration $c(x, y, z, t)$ is gathered in a single area and does not change much over time. Its main difficulty lies in fine-tuning the trajectory such that it follows bands of high concentration within the general area of the CB. Both extremes of $\boldsymbol{s}(t)$ are placed at the same point ($[-750, -600, -0.1]$ m) on the left of the lake, and the start time $t_0^{mission} = 65$ h after the start of the simulation. Representative solutions obtained by the planner for this scenario are shown in Fig. 5a.

Scenario B. It uses Simulation B, which has two areas with high cyanobacterial concentration. The planner needs to adjust the trajectory so that both are visited in a direct and timely manner. To reduce the ASV displacement time during the mission and to showcase that the initial and final points can differ, the initial trajectory location $\boldsymbol{s}(t_0)$ is set in the north of the map at $[380, 100, -0.1]$ m, while the final location is set in the south at $[-100, -450, -0.1]$ m. Finally, $t_0^{mission} = 54$ h. Figure 5b shows representative solutions.

Scenario C. It uses Simulation C, where the particle positions change fast while they follow the main current of the flow, providing a challenge for the planner to adjust the temporal variable and handle the velocity constraint. The ASV initial and final locations (placed closer to the CB at $[300, -550, -0.1]$ m) coincide again and $t_0^{mission} = 6$ h. Figure 5c shows representative solutions.

4.2 Planner Configurations

For all the scenarios, the planner is configured with a population of 100 individuals (N_s) and 90 children (N_c). It is also run over 500 generations (N_g), and the crossover probability $p_{xover} = 80\%$. Regarding the remaining parameters, for comparison purposes, two variants of the algorithm are executed for each scenario: one with refinement (called V1 herafter) starting with 5 spline nodes ($N_f^0 = 4$) and ending with 12 nodes ($N_f^{end} = 11$), and one without refinement (V2) that starts and ends with 12 nodes ($N_f^0 = N_f^{end} = 11$).

4.3 Comparative Analysis Description

Due to the stochastic nature of the NSGA-II, we characterize the performance of our planner for each scenario by statistically analysing [5] the results obtained by 120 executions of each variant of the algorithm (V1 and V2) during its 500 running generations[10].

To compare the performance of V1 and V2, the versions of the planner with and without refinement the best value of each OF is extracted from the population for each execution and generation. The best Pareto front at each generation is also extracted to calculate the HyperVolume Estimation (HVE) as in [4], using for the Monte-Carlo samples the ranges $[0, T_{max}^{mission}]$ for OF_1, $[0, 10^4]$ for OF_2, and $[0, A_c \sup_t(N_p(t))]$ for OF_3. The mean of these variables over the 120 executions and the 95% confidence interval are represented for each scenario in the four top plots of Fig. 4 using a colored line and share area (red for V1 and green for V2) against the execution time[11]. As HVE and OF_3 are subject to maximize, higher is better for them, while the opposite is true for OF_1 and OF_2, which are subject to minimize. Each plot also includes blue vertical dashed lines when a refinement was made in V1. The bottom plots represent in a color map the percentage of the runs of V1 (top row) and V2 (bottom row) that have obtained at least one solution that meets all CFs at every 10 generations of the algorithm (using green when all runs have found feasible solutions).

Finally, for each scenario and planner variant, the solution of the final generation with the best value of OF_3 is selected (according to the operator preferences) and represented d in Fig. 5, using three different plots for each scenario. The one on the top shows a map of the lake, and the color scale represents the average CC concentration on each water column throughout the longest mission. The white line denotes the limits of the navigable zone (with the recreational area on the right), and the red/green line represents the horizontal components $(s_x(t), s_y(t))$ of the solution corresponding to V1/V2. The beginning of the trajectories is marked with a cross and the end with a circle, while the nodes of the spline are represented with dots. On the bottom two plots, we represent in the vertical axis the vertical component $(s_z(t))$ of the previous two horizontal trajectories using the same markers as before, while the time since the mission start is represented in the horizontal axes. In this case, the color scale shows the CB concentration at different depths on the water column under the ASV.

4.4 Discussion

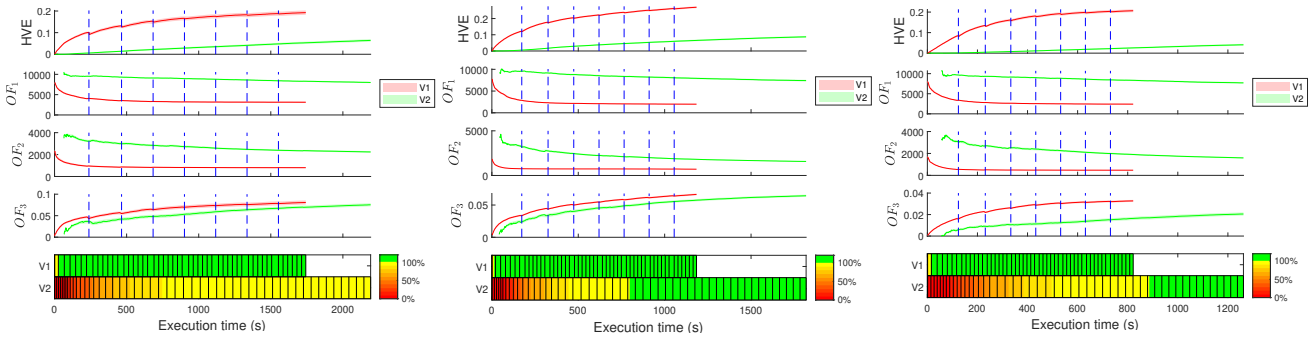
Figure 4 shows that the refinement included in V1 can significantly speed up the search and improve the solutions found by our planner. In particular, it displays how HVE, OF_1, OF_2 and OF_3 obtained by V1 improve more and converge faster than their counterparts from V2. Besides, the feasibility plots show that V1 finds feasible solutions at all its runs almost instantaneously (which allows to improve OFs quickly from the initial generations), while V2 takes longer to find feasible solutions (improving later and much slower the OFs).

The solutions displayed in Fig. 5 illustrate similar facts. The representative trajectory found by V1 (in red) ends significantly quicker, is more direct, and tracks better areas with higher CB concentration. On the contrary, the trajectories found by V2 (in green) have more room for improvement since significant portions are superfluous and could be eliminated without negatively impacting the quality of the measurements taken during the actual missions.

Figure 5 also shows that the goals set for each scenario are met by V1. In Scenario A, the ASV trajectory adequately adjusts to the center of the bands of lower CB concentration and does not spend useless time in high-concentration areas. In Scenario B, the ASV visits both areas of high concentration with a direct, quick route.

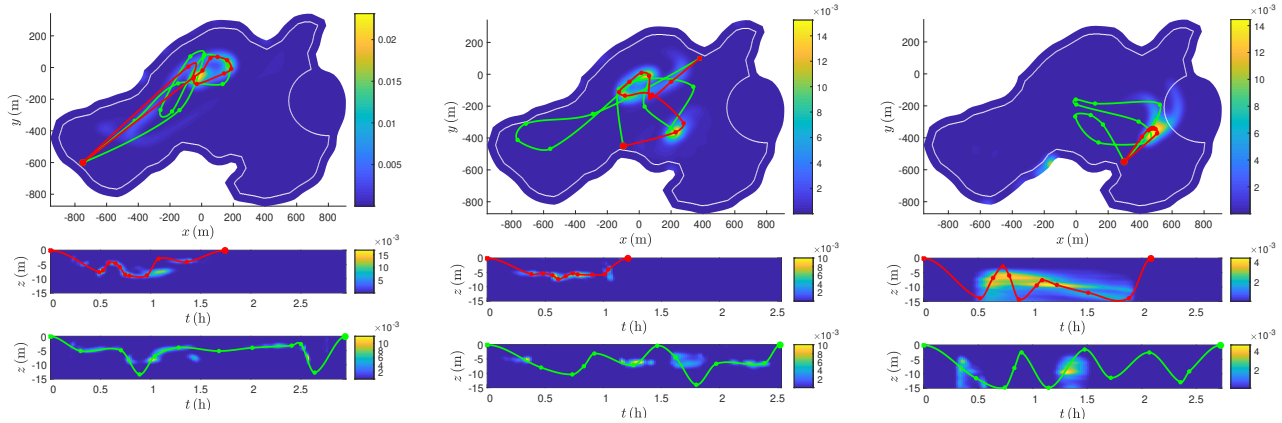
[10] To be able to reconstruct any solution of the planner and perform the statistical analysis, we keep a log of the decision variables of the entire population for for each variant, execution and generation.

[11] The computation times are obtained when the planner is run in Matlab R2019a and executed inside an Intel Core i7-6700HQ CPU (2.60 GHz) with 16 GB of RAM.



(a) Scenario A (b) Scenario B (c) Scenario C

Figure 4: Evolution of the HVE and objective functions



(a) Scenario A (b) Scenario B (c) Scenario C

Figure 5: Selected solutions provided by the planner

And in Scenario C, the ASV captures the fast-moving CB areas without going into the non-navigable zone when the CB enters it. In short, in all the scenarios, the planner finely adjusts the probe's vertical position to capture the CBs during their vertical migration.

5 CONCLUSIONS

This paper presents an EA-based planner for ASV missions in lentic waters that generates trajectories defined by splines that simultaneously optimize three objectives: the mission duration, the trajectory length, and the CB concentration along the ASV trajectory. To this end, the planner optimizes the spline nodes with NSGA-II (a proven multi-objective optimization method in path planning) and is supported by a physical simulation of the evolution of the CBs. Restrictions added to the planner ensure the feasibility and safety of the trajectory from the perspective of the ASV and the onboard sensor. Finally, it is worth noting that the planner includes a refinement operator that periodically increases the number of decision variables and helps it to find overall good solutions quicker than when the operator is disabled.

Although the results obtained with the planner already prove its capability to get overall good trajectories, there is still room for improvement. In particular, a convergence check could be added to the algorithm, to be used as a stop condition and to determine when to perform the refinements. We have to explore also the effects of the parameters on the solution; and the use of different kinds of splines, mutations, placements for the refinement nodes, objective functions related to the CB concentration and EA optimizers (e.g. NSGA-III [10]). Some improvements could also be made to the simulator, such as taking a Monte Carlo approach or using continuous media methods for the simulation.

ACKNOWLEDGMENTS

This work has been supported by the Research Projects IA-GES-BLOOM-CM (Y2020/TCS-6420) funded by the Synergic program of the Comunidad Autónoma de Madrid, SMART-BLOOMS (TED2021-130123B-I00) by MCIN/AEI/10.13039/501100011033 and the European Union NextGenerationEU/PRTR, and INSERTION (PID2021-127648OB-C33) by the Knowledge Generation program of the Spanish Ministry of Science and Innovation. Gonzalo Carazo-Barbero's research is specifically supported by the PhD Grant CT82/20-CT83/20 of the Universidad Complutense of Madrid and Banco Santander.

REFERENCES

[1] E. Aparicio-Medrano. 2014. *Physical aspects explaining cyanobacteria scum formation in natural systems.* Ph. D. Dissertation. Eindhoven.

[2] M. Arzamendia, D. Gregor, D.G. Reina, and S.L. Toral. 2019. An evolutionary approach to constrained path planning of an autonomous surface vehicle for maximizing the covered area of Ypacarai Lake. *Soft Computing* 23 (2019), 1723–1734.

[3] M. Arzamendia, D. Gregor, D. G. Reina, S. L. Toral, and R. Gregor. 2016. Evolutionary Path Planning of an Autonomous Surface Vehicle for Water Quality Monitoring. In *Int. Conf. on Developments in e-Systems Engineering.*

[4] J. Bader, K. Deb, and E. Zitzler. 2008. Faster Hypervolume-Based Search Using Monte Carlo Sampling. In *International Conference on Multiple Criteria Decision Making.*

[5] E. Besada-Portas, L. de la Torre, A. Moreno, and J.L. Risco-Martín. 2013. On the performance comparison of multi-objective evolutionary UAV path planners. *Information Sciences* 238 (2013), 111–125.

[6] S.R. Bistafa. 2017. On the development of the Navier-Stokes equation by Navier. *Revista Brasileira de Ensino de Física* 40, 2 (2017).

[7] P. Bouguer. 1729. *Essai d'optique sur la gradation de la lumière [Optics essay on the attenuation of light] (in French).* Claude Jombert. 16–22 pages.

[8] G. Carazo-Barbero, E. Besada-Portas, J. M. Girón-Sierra, and J. A. Lopez-Orozco. 2021. EA-based ASV Trajectory Planner for Pollution Detection in Lentic Waters. In *EvoApplications 2021.*

[9] COMSOL [n. d.]. COMSOL Multiphysics Webpage. https://www.comsol.com/. accessed on 31/01/2021.

[10] K. Deb and H. Jain. 2014. An Evolutionary Many-Objective Optimization Algorithm Using Reference-Point-Based Nondominated Sorting Approach, Part I: Solving Problems With Box Constraints. *IEEE Transactions on Evolutionary Computation* 18, 4 (2014), 577–601.

[11] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation* 6, 2 (2002).

[12] European Comission [n. d.]. European Commission drinking water directive. https://www.europeandrinkingwater.eu/initiative/dwd-2021/. accessed on 31/01/2023.

[13] S. Ghambari, M. Golabi, J. Lepagnot, M. Brévilliers, L. Jourdan, and L. Idoumghar. 2020. An Enhanced NSGA-II for Multiobjective UAV Path Planning in Urban Environments. In *2020 IEEE 32nd International Conference on Tools with Artificial Intelligence (ICTAI).* 106–111. https://doi.org/10.1109/ICTAI50040.2020.00027

[14] D. E. Goldberg and K. Deb. 1991. A Comparative Analysis of Selection Schemes Used in Genetic Algorithms. Foundations of Genetic Algorithms, Vol. 1. Elsevier, 69–93. https://doi.org/10.1016/B978-0-08-050684-5.50008-2

[15] M. Janga-Reddy and D. Nagesh-Kumar. 2020. Evolutionary algorithms, swarm intelligence methods, and their applications in water resources engineering: a state of the art review. *H2Open Journal* 3, 1 (2020).

[16] M. J. T. Kathen, I. J. Flores, and D. G. Reina. 2021. An informative path planner for a swarm of ASVs based on an enhanced PSO with Gaussian surrogate model components intended for water monitoring applications. *Electronics* 10, 13 (2021).

[17] M. J. T. Kathen, I. J. Flores, D. G. Reina, and A. T Córdoba. 2021. Autonomous monitoring system for water resources based on PSO and Gaussian process. In *IEEE congress on evolutionary computation.*

[18] Z. Liua, Y. Zhanga, X. Yua, and C. Yuana. 2016. USVs: An overview of developments and challenges. *Annual Reviews in Control* 41 (2016).

[19] MATLAB [n. d.]. MATLAB Webpage. https://www.mathworks.com/. accessed on 31/01/2021.

[20] J. Meriluoto, L. Spoof, and G.A. Codd. 2017. *Handbook of cyanobacterial monitoring and cyanotoxin analisis.* Wiley.

[21] M. Panda, B. Das, B. Subudhi, and B.B. Pati. 2020. A Comprehensive Review of Path Planning Algorithms for Autonomous Underwater Vehicles. *International Journal of Automation and Computing* 17 (2020), 321–352.

[22] F. Peralta, D. G. Reina, and S. Toral. 2023. Water quality online modeling using multi-objective and multi-agent Bayesian Optimization with region partitioning. *Mechatronics* 91 (2023).

[23] S. Perez-Carabaza, E. Besada-Portas, J. A. Lopez-Orozco, and J. M. de la Cruz. 2016. A Real World Multi-UAV Evolutionary Planner for Minimum Time Target Detection. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO).*

[24] S. Pérez-Carabaza, E. Besada-Portas, J. A. Lopez-Orozco, and G. Pajares. 2019. Minimum Time Search in Real-World Scenarios Using Multiple UAVs with Onboard Orientable Cameras. *Journal of Sensors* 2019 (2019).

[25] M. R. C. Qazani, M. Karkoub, H. Asadi, C. P. Lim, A. W-C Liew, and S. Nahavandi. 2022. Multi-objective NSGA-II for Weight Tuning of a Nonlinear Model Predictive Controller in Autonomous Vehicles. In *IEEE International Conference on Systems, Man, and Cybernetics (SMC).* 2820–2826.

[26] M. H. Ranjbar, D. P. Hamilton, A. Etemad-Shahidi, and F. Helfer. 2021. Individual-based modelling of cyanobacteria blooms: Physical and physiological processes. *Science of The Total Environment* 792 (2021), 148418.

[27] A. Ravankar, A.A. Ravankar, Y. Kobayashi, Y. Hoshino, and C. Peng. 2018. Path Smoothing Techniques in Robot Navigation: State-of-the-Art, Current and Future Challenges. *Sensors* 18 (2018).

[28] B.Z. Rousso, E. Bertone, R. Stewart, and D.P. Hamilton. 2020. A systematic literature review of forecasting and predictive models for cyanobacteria blooms in freshwater lakes. *Water Research* 182 (2020).

[29] S. Sabino and A. Grilo. 2019. NSGA-II based Joint Topology and Routing Optimization of Mesh Networks with Flying Access Points. *Procedia Computer Science* 160 (2019), 165–172. The International Conference on Emerging Ubiquitous Systems and Pervasive Networks (EUSPN-2019).

[30] I. J. Schoenberg. 1973. *Cardinal spline interpolation.* SIAM.

[31] United Nations [n. d.]. 2030 Agenda for Sustainable Development. https://sdgs.un.org/goals. accessed on 15/10/2020.

[32] G. Xia, Z. Han, B. Zhao, C. Liu, and X. Wang. 2019. Global Path Planning for Unmanned Surface Vehicle Based on Improved Quantum Ant Colony Algorithm. *Mathematical Problems in Engineering* 2019 (2019).

[33] C. Xiong, D. Chen, D. Lu, Z. Zeng, and L. Lian. 2019. Path planning of multiple autonomous marine vehicles for adaptive sampling using Voronoi-based ant colony optimization. *Robotics and Autonomous Systems* 115 (2019), 90 – 103.

[34] C. Xiong, H. Zhou, D. Lu, Z. Zeng, L. Lian, and C. Yu. 2020. Rapidly-Exploring Adaptive Sampling Tree*: A Sample-Based Path-Planning Algorithm for Unmanned Marine Vehicles Information Gathering in Variable Ocean Environments. *Sensors* 20 (2020).