DEVS-BASED MODELING AND SIMULATION OF DATA-DRIVEN EXPLORATION ALGORITHMS OF LENTIC WATER BODIES WITH AN ASV

Samuel Ferrero-Losada
Eva Besada-Portas
José L. Risco-Martín
José A. López-Orozco

Department of Computer Architecture and Automation
Universidad Complutense de Madrid
Calle Prof. José García Santesmases, 9
28040 Madrid, SPAIN
{saferr03,ebesada,jlrisco,jalo}@ucm.es

ABSTRACT

This paper presents a Discrete Event System Specification (DEVS) modular architecture for modeling and simulating the behavior of an Autonomous Surface Vehicle (ASV), guided by gradient-free optimization and contour detection algorithms, while searching the extreme points and constant curves of a variable of interest in a water body. The ASV model integrates its dynamic differential equations and is controlled by a Guidance, Navigation, and Control (GNC) system implemented through several atomic models to facilitate changes in the behavior of any of them and test different types of regulators and guidance approaches in the future. The paper results show that the system is working correctly for the already supported missions (seeking extreme points and contour curves) and that adapting the specification to each of them is possible by modifying one of the models of the GNC.

Keywords: Data-driven Exploration, Water-quality Monitoring, DEVS, ASVs, GNCs.

1 INTRODUCTION

In recent years, water availability and management have become increasingly pressing, affecting not only humans but also other species living around and within water bodies. The problem is so relevant that it is listed as the third goal of the United Nations 2030 Agenda for Sustainable Development (United Nations 2023). It is being addressed through novel water measurements and actions, included in recent international regulations (e.g., United States Environmental Protection Agency 2023 and European Comission 2023), which should lead to new water monitoring and management strategies.

Some of the current monitoring methods are manually taking samples from boats or automatically collecting data from probes placed in fixed geographical positions. Combining the benefits of both approaches by using Autonomous Surface Vehicles (ASVs, a type of robotized boats) capable of moving the probes seems a logical/natural improvement. Under this approach, automatic probes can be taken to the points of interest of a water body, making it possible to undertake faster and more detailed water studies (Hitz et al. 2012, Siyang

and Kerdcharoen 2016, Shuo et al. 2017, Carazo-Barbero et al. 2021, Giron-Sierra and Chacon-Sombria 2021). To achieve it, it is necessary to develop a complete automation system for the ASV that decides how to make the ASV act in each situation, including advanced location, planning, guidance, navigation, control, and detection techniques (Liu et al. 2016). Moreover, with such a system in charge of automating ASVs working in real-world environments, growing complexity will benefit from its division into small interconnected tasks, implemented under well-established Systems Engineering approaches.

At the same time, modeling and simulation of the ASVs, its automation system, and the environment become particularly important to understand each part of the system better, prevent errors, and discover new aspects of the real-world cases under study. Model-Based Systems Engineering (MBSE) strategies are helpful in adequately managing the simulation of a system that comprises many different models (e.g., for the ASVs, probes, and environment conditions) and tasks of the automation system. A large group of engineers or researchers often develop these tasks, usually divided into small groups dedicated to specific parts of the project. MBSE provides the backbone where all these elements can coexist and interact. In this context, the DEVS (Discrete Event System Specification, Zeigler 2019) formalism can help to deploy MBSE-based solutions, as it provides a robust discrete event modeling and simulation framework for complex systems. DEVS can explicitly define models' structure and behavior, and rigorously separates the model from its simulation. Systems modeled under this formalism take advantage of DEVS completeness, verifiability, extensibility, and maintainability capabilities. In particular, our system is modeled with the Python branch of xDEVS, a cross-platform and DEVS application programming interface capable of running sequential or parallel simulations with good performance (Risco-Martín et al. 2022).

To show the benefits of using DEVS for developing and simulating the behavior of an automation system for ASVs, we use this formalism for implementing a data-driven exploration Guidance, Navigation, and Control (GNC) system. A GNC lets an ASV determine the location of the extreme points or a given contour level of one of the variables of interest of a water body. In particular, this work is inspired by the approaches presented in Besada-Portas et al. (2021), which exploit the Simplex Optimization Algorithm by Nelder and Mead (1965) and the path following algorithm by Mezher and Philippe (2000) in order to determine the locations (waypoints) where the ASV has to take the following measurements of the variable of interest (whose values are unknown before hand). Besides using DEVS to implement and simulate the type of GNC presented in Besada-Portas et al. (2021), we also apply the selected approach over a catamaran with two propellers (instead of over a single hull boat with one propeller and a rudder). We also slightly modified the ASV course controller, moving the ASV to each new waypoint instead of forcing it to follow the straight line defined between two consecutive waypoints.

Finally, it is worth noting that the work presented in this paper is not the first one where DEVS is used to tackle problems involving autonomous vehicles or trajectories analysis. For instance, Holman et al. (2010) propose a *Cell-DEVS*-based path-planner for Unmanned Aerial Vehicles (UAVs); Moreno et al. (2011) introduce a DEVS-based model to evaluate the trajectories of multiple UAVs performing missions in hostile environments; and Bordón-Ruiz et al. (2021) present two DEVS-based evaluators for trajectories of UAVs performing target-search missions. Although we have only named a few works, this selection already illustrates the benefits of exploiting DEVS for these problems.

2 PROBLEM DESCRIPTION AND FORMULATION

This section describes the water variable-of-interest characterization problem with an ASV equipped with a probe, considering two different types of missions: 1) locating the points with extreme values of the variable of interest and 2) determining a contour (level curve) for a given value of the variable of interest.

In more detail, we assume in both types of missions that the ASV can freely move (only restricted by its dynamics and the GNC characteristics) defining a 2D trajectory $\boldsymbol{p}(t) = [x(t), y(t)]$ over the water surface,

starting in a given position $\boldsymbol{p}(0)$ of the water body. The course and shape of the ASV trajectory are defined during the mission by: (i) the ASV's dynamics, (ii) the measurements taken by the probe at each ASV location, and (iii) the decisions taken by the GNC to determine the extreme point or a level curve of the variable of interest at the water surface. To simplify the problem, the behavior of the probe is simulated by a deterministic static function $f(\boldsymbol{p}(t))$. This function must be provided for each scenario before its simulation takes place, as it lets the ASV measure the variable of interest at any given point $\boldsymbol{p}(t)$ of its trajectory.

The following two sections detail the ASV dynamics model and the guidance subsystem behavior. In the third one, we summarize the main characteristics of the remaining elements of our system.

2.1 ASV Dynamics Model

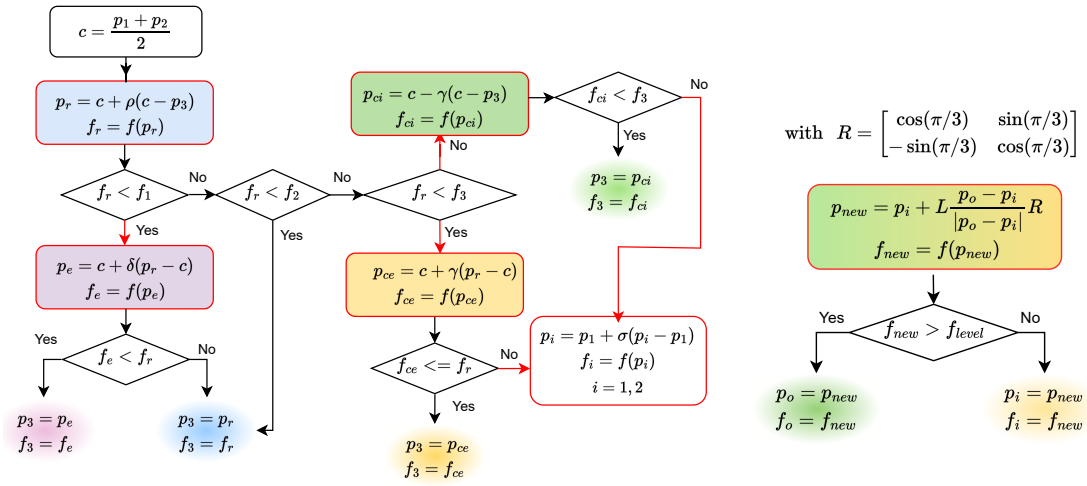
The ASV of this paper is a catamaran with two propellers, and the mass (m), height (H), width (W), and length (L) stated at the first column of Table 1. At every moment t, the ASV state is $\boldsymbol{s}(t) = [x(t), y(t), \varphi(t), u(t), v(t), r(t)]$, where $[x(t), y(t)]$ represent the ASV location $\boldsymbol{p}(t)$, $\varphi(t)$ its facing angle, $[u(t), v(t)]$ its longitudinal and transversal velocities, and $r(t)$ its rotation speed. The ASV behavior is modeled with the differential equations in the third column of Table 1 and integrated with the 4th-order Runge-Kutta method. The ASV dynamics do not currently consider environmental effects, such as wind or water currents. The forces that alter the ASV state are frictions and the thrusts (T_i with $i \in \{L, R\}$) provided by the left (L) and right (R) propellers, which are controlled by the PWM actions/signals a_i that are provided to each drive. The PWM-to-Thurst curve, modeled by relation (R1) of Table 1, makes use of different second degree polynomial for the forward (first line), backward (second line) or dead zone region (third line) of the PWM signal. It is parameterized according to the observed behavior of our ASV. Besides, relation (R2) determines the total forces (X, Y) over the ASV in the body-axis obtained, taking into account the thrusts, the longitudinal and transversal velocities, and the friction coefficients (c_{front} and $c_{sideways}$). At the same time, relation (R3) calculates the ASV torque (N) considering the thrusts, the ASV rotation speed, the friction coefficient (c_{rotate}), the distance (d_{hel}) from the ASV Mass Center (MC) to the propellers, and the angle (θ) between the ASV longitudinal axis and the vector that joins the MC and each propeller.

2.2 Guidance Subsystem Behavior

The guidance subsystem of the GNC is responsible for obtaining the next waypoint ($\boldsymbol{p}_k$) that the ASV should visit, according to the measurements of the probe at certain locations of the water body and to the intended objective of the mission: determine extreme locations or the contour curves of the variable of interest. Although the approaches used to implement both types of missions are already introduced in Besada-Portas

Table 1: ASV Dynamics model equations and parameters.

ASV parameters	Inputs, states and intermediate variables relationships	Differential Eq.
$m = 40\ Kg$ $H = 0.98\ m$ $L = 0.73\ m$ $W = 0.70\ m$ $d_{hel} = 0.399\ m$ $c_{front} = 1.629$ $c_{sidways} = 29.71$ $c_{rotate} = 25.12$	$T_i = \left\{ \begin{array}{l} (1.3e^{-5} \cdot a_i^2 - 3.4e^{-2} \cdot a_i + 21.5)\ if\ a_i \geq 1540 \\ 9.8 \cdot (9.2e^{-6} \cdot a_i^2 - 3.2e^{-2} \cdot a_i + 27.1)\ if\ a_i \leq 1460 \\ 0 \qquad\qquad\qquad\qquad if\ 1460 \leq a_i \leq 1540 \end{array} \right\}$ (R1) $\left. \begin{array}{l} X = -\ c_{\text{front}} \cdot u\ +\ (T_L\ +\ T_R) \\ Y = -\ c_{\text{sideways}} \cdot v \end{array} \right\}$ (R2) $N = -\ c_{\text{rotate}} \cdot r\ +\ (T_R - T_L) \cdot d_{hel} \cdot sin(\theta)$ (R3)	$\dot{x} = u\cos\varphi - v\sin\varphi$ $\dot{y} = u\sin\varphi - v\cos\varphi$ $\dot{\varphi} = r$ $\dot{u} = \frac{X}{m}\ +\ r \cdot v$ $\dot{v} = \frac{Y}{m}\ -\ r \cdot u$ $\dot{r} = \frac{N}{\frac{m}{12} \cdot \left(L^2 + W^2\right)}$



(a) Nelder-Mead Simplex Optimization Algorithm. (NMSOA) (b) PAT Algorithm.

Figure 1: Guidance algorithms.

et al. (2021), we describe them again in the following sections to help the reader understand later the corresponding DEVs models of this subsystem.

2.2.1 Guidance Subsystem for Extreme-points Seeking

To let the ASV determine an extreme point of the variable of interest, the guidance module uses the Nelder-Mead Simplex Optimization Algorithm (NMSOA, Nelder and Mead 1965), which triangulates the search space (water body surface in our case) in order to find a minimum of a given function (whose sign can be changed to look for a maximum). The number and type of operations involved in each case depend on the results of the comparisons of the function values, measured in successive points of the triangles.

To carry out the optimization process, this subsystem evaluates the function, initially in three unaligned points ($\boldsymbol{p}_1$, $\boldsymbol{p}_2$ and $\boldsymbol{p}_3$), and iteratively in some intermediate ones ($\boldsymbol{p}_r$, $\boldsymbol{p}_e$, $\boldsymbol{p}_{ce}$ and $\boldsymbol{p}_{ci}$), obtained according to the expressions and the algorithm schematized in Fig. 1(a) and to its parameters (ρ, γ, δ and σ). However, to compare the variable of interest in the intermediate waypoints, the ASV must first arrive at each before making any other calculations to obtain new destinations. For this reason, this subsystem outputs a set of three waypoints initially or only one during the iterations of the algorithm. Moreover, the red-framed boxes of Fig. 1 indicate the cases where the algorithm must provide the waypoint calculated within the box to the navigation subsystem and wait until the ASV can measure this location before continuing. Once the decision tree of the algorithm reaches an ending leaf, the points of the original set (i.e. $\boldsymbol{p}_1$, $\boldsymbol{p}_2$ and $\boldsymbol{p}_3$) are updated, sorted from the lowest to the highest of their function values. The process is re-started until the triangle defined by the last set of $\boldsymbol{p}_1$, $\boldsymbol{p}_2$, and $\boldsymbol{p}_3$ is small enough to consider that it has collapsed around an extreme point (i.e., when the area defined by these three points is smaller than d_{area}).

2.2.2 Guidance Subsystem for Determining Contour Curves

This mission requires the ASV to initially find a pair of points situated *i*nside ($\boldsymbol{p}_i$) and *o*utside ($\boldsymbol{p}_o$) the contour curve under analysis. Then, the PATh following algorithm (PAT, Mezher and Philippe 2000) comes into play, choosing new destinations among the nodes of a grid of equilateral triangles of side T placed over the water surface. This grid is built as the ASV advances and as Fig. 1(b) shows, by adding a new

vertex opposite to the side defined by $\boldsymbol{p}_i$ and $\boldsymbol{p}_o$ of the last triangle, proposing it as the new waypoint for the navigation subsystem, and testing if this new waypoint falls inside or outside the contour curve. The process ends when the last waypoint is closer than d_{stop} to one of the two first waypoints, closing the contour.

2.3 Remaining Models and Subsystems

This section sketches the functionality of the remaining elements of the GNC and some additional functionality-supporting models:

- The Navigation subsystem receives from the ASV Dynamics model the ASV state $\boldsymbol{s}(t)$ and from the Guidance module the next waypoint $\boldsymbol{p}_k$ that the ASV must visit. With that information, it calculates the setpoint course (ϕ_{sp}) and setpoint speed (v_{sp}) that the ASV must reach and sends them to the Controller. In particular, ϕ_{sp} is calculated as the angle of the vector from $\boldsymbol{p}(t)$ to $\boldsymbol{p}_k$, and v_{sp} is reduced from v_{sp}^{high} to v_{sp}^{low} as the ASV gets close enough to the current $\boldsymbol{p}_k$ (i.e. when $|\boldsymbol{p}(t)-\boldsymbol{p}_k| < d_{speed}$). This subsystem also sends a *reset* signal to the Controller when reaching any $\boldsymbol{p}_k$ (i.e., when $|\boldsymbol{p}(t)-\boldsymbol{p}_k| < d_{reach}$) and a *request* to the Guidance subsystems when reaching the last received $\boldsymbol{p}_k$.
- The Controller translates the setpoints $[\phi_{sp}, v_{sp}]$ from the Navigation subsystem into high-level control signals for the Transducer, taking into account the rotation and acceleration restrictions of the ASV. To this end, this subsystem first obtains a rotation control signal (c_{rot}) with a Bounded Proportional Integral with Anti-Windup (BPI-AW) regulator that 1) takes into account the error (difference) between the actual orientation of the ASV $\phi(t)$ and the setpoint course ϕ_{sp}, and that 2) bounds its output c_{rot} within $[-21,21]/9.8$. Afterward, this subsystem obtains a velocity control signal (c_v) with a specialized BPI-AW regulator that 1) takes into account the error between the actual speed of the ASV $\left(\sqrt{u(t)^2+v(t)^2}\right)$ and the setpoint velocity v_{sp}, that 2) reduces its output accordingly to c_{rot} and the bounding limits $[0, 7.4]$, and that 3) smooths c_v with a first-order low-pass filter. Note that c_v is reduced accordingly to c_{rot} to adjust the ASV speed to its rotation and that the low-pass filter's purpose is to reduce steep velocity variations.
- The Transformer model translates the high-level control signals $[c_{rot}, c_v]$ from the Controller into the action signals $[a_L, a_R]$ for the propellers of the ASV Dynamics model. To this end, it distributes the controller effort between both propellers, taking into account the inverse of the PWM-Thrust curves, which are already being used in the ASV Dynamics model to convert the PWM signals into Thrusts.
- The Registry and Visualization models cover the secondary functions of storing together and plotting in graphs simulation data and parameters.

3 DEVS MODELING

This section presents the architecture and behavior of our DEVS models for simulating these real-world ASV exploration missions. The architecture and majority of the models' behavior will be common to the two mission types, differing mainly in the behavior of the Guidance and Visualization models. The rest of the models remain unchanged since they have been defined as independent from the others as possible. Our system exploits the versatility offered by DEVS to encapsulate the behavior of different parts of a system in different models to build complex, easy-to-adapt, and scalable systems.

The general architecture of our specification is shown in Fig. 2, which also includes, at its top-left, the specification of the extreme-point seeking guidance subsystem (the only coupled model of the specification). Each DEVS model is described in the following sections, starting with the ones common to both missions.

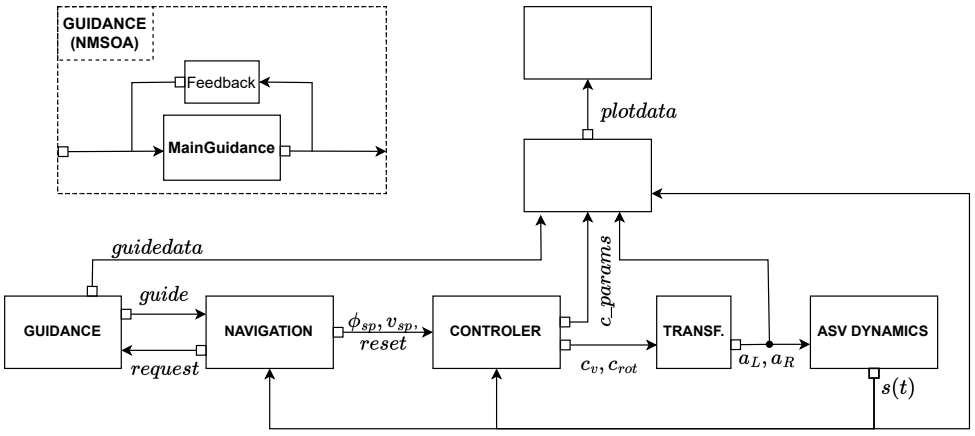


Figure 2: General architecture of our specification and of our DEVS Extreme-point Seeker Guidance Model.

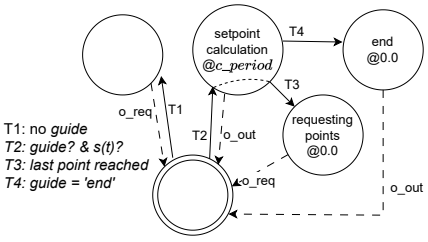
3.1 ASV Dynamics Model

The ASV Dynamics model simulates the ASV displacement and effectively initializes and moves the simulation forward through a recurring internal transition. With each iteration, the ASV state $\boldsymbol{s}(t)$ is updated and outputted through a unique port to the Navigation, Controller, and Registry models. The simulation runs periodically, uninterrupted until the end of the simulation, using an invariant time advance equal to the sample step required by the 4th Order Runge-Kutta method that integrates the differential equations presented in the third column of Table 1. The PWM signals $[a_L, a_R]$ from the Transformer are always updated through the external transition, and used later in the internal one. The model becomes passive after receiving the 'end' signal. Finally, as this model boots up the others, it has to receive an initial state $\boldsymbol{s}(0)$ and PWM signal $[a_L, a_R]$. By default they are $\boldsymbol{s}(0) = [x(0), y(0), 0, 0, 0, 0]$ and $[a_L, a_R] = [1460, 1460]$, which makes the ASV remain at $\boldsymbol{p}(0)$ until a different PWM signal $[a_L, a_R]$ is received.

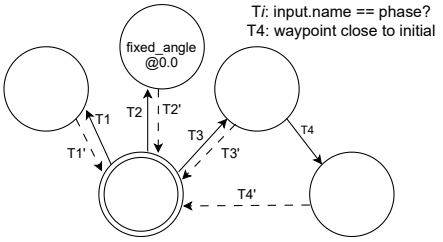
3.2 Navigation Model

This is the second model that takes part in the process. After the ASV Dynamics model receives the initial state $\boldsymbol{s}(0)$, the Navigation model sends a request to the Guidance model for an initial set of waypoints $\boldsymbol{p}_k$. The waypoints are accompanied by a *name* that indicates the current stage of the mission process (e.g., if the ASV is targeting an initial waypoint or any of $\boldsymbol{p}_{e,r,ce,ci}$). When the last waypoint stored in the Navigation model is reached, its *name* is sent back with the current ASV location $\boldsymbol{p}(t)$ and its associated function value $\boldsymbol{f}(\boldsymbol{p}(t))$. Hence, the subsequent operations of the Guidance model can consider $\boldsymbol{p}_k = \boldsymbol{p}(t)$.

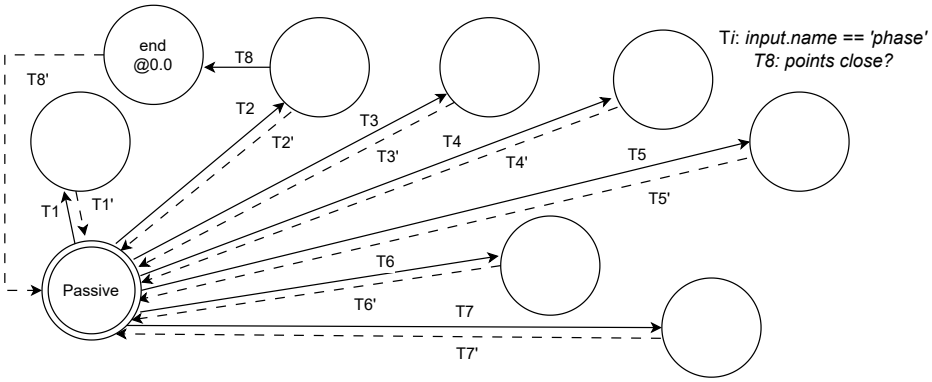
The Navigation model has an input port for receiving waypoints $\boldsymbol{p}_k$ and two output ports: *o_out* to send setpoints $[\phi_{sp}, v_{sp}]$ and *reset* to the Controller model; and *o_req* to request $\boldsymbol{p}_k$ to the Guidance model. The behavior of the Navigation is presented in Fig. 3(a), where continuous/discontinuous lines represent external/internal transitions. Internal transitions make the model passive once the required action is performed, calculating the new $[\phi_{sp}, v_{sp}]$ or requesting new waypoints $\boldsymbol{p}_k$. External transition labels indicate the conditions needed to trigger them, while internal transition labels indicate the output ports. Each state shows its name and the time of advance (@xx) required to complete it. The states with @0.0 are instantaneous and carry out streamlined operations as fast as possible. The advance time *c_period* of the state *setpoint calculation* establishes the time separation between $[\phi_{sp}, v_{sp}]$ updates, making this state avoid new calculations while it is 'busy'. If during the setpoints calculation, the model finds out that there are no more waypoints to reach, it triggers transition T3 to request new ones.



(a) Navigation model behavior.



(b) Guidance model behavior for determining contours.



(c) MainGuidance model behavior for seeeking extreme points.

Figure 3: States Diagrams of the behavior of some of the models of the system.

The *setpoint calculation* state makes slightly different operations for each type of mission. While seeking extreme-points, it follows the default process explained in section 2.3 to reach the waypoints. Two different situations come into play for determining contours: first, the ASV must find the targeted contour, and afterward, it can start using the PAT algorithm and the default process to reach its waypoints. However, to achieve the first goal, the ASV must move forward in a straight line until it consecutively registers one value inside and another outside the contour. Hence, for this particular case, ϕ_{sp} is fixed to the angle of the straight line the ASV must follow.

3.3 Controller and Transformer Models

These two models have a similar behavioral definition, as they have only one active state and are always sequentially activated. On the one hand, when the Controller model receives new setpoints, it calculates their corresponding high-level control signals. It sends them to the Transformer model and the Registry model (accompanied, in this case, by some of the internal values of the BPI-AW regulators). On the other hand, the Transformer model is activated when it receives the setpoints from the Controller model, makes the appropriate calculations, and outputs the PWM actions to the ASV Dynamics and Registry models.

3.4 Registry and Visualization Models

These models have a more supportive function. The Registry model receives all relevant data and parameters of the simulation and stores them together to analyze them quickly afterward. It also sends the most pertinent information to the Visualization model, which processes it and generates different types of plots (such as

those presented in Fig. 4 of Section 4). Both models have only one 'active' state (respectively, for storing data and plotting them), which in the Registry model is activated every time it receives an input, while in the Visualization model is activated when the simulation is ending.

3.5 Guidance Model for Extreme-points Seeking

This Guidance model is a streamlined coupled DEVS model, represented at the top-left of Fig. 2. This structure is used because there are some situations where it is necessary to input the output of the previous state to the Guidance model itself. A straightforward way of doing it is using a Feedback model with the only purpose of receiving the output of a second model, called MainGuidance hereafter and implementing the remaining functionality of this Guidance model, and sending it back to it instantly.

The behavior of MainGuidance is conceptually similar to that of the Control and Transformer models: it is only activated when some operation is required and becomes passive after every output. This is illustrated in Fig. 3(c), where most states become active by external transitions from the passive state and return to it after an internal transition. The condition deciding which state to activate is the name of the input triggering the external transition. Note also that these states have a null advance time to return the calculated waypoints $\boldsymbol{p}_k$ to the Navigation model as soon as possible.

All the states of MainGuidance, except the 'initial' and 'end' ones, relate to consecutive steps of NMSOA. Generally, at the start of every iteration, the activated state is 'loop_start', followed by 'simplex_xr'. Next, depending on the value measured in the last waypoint, one of 'simplex_xe/xci/xce' might follow. And afterward, depending on the previous and new measurements, the state 'loop_end' may or may not be activated. Finally, a new iteration will start again with 'loop_start'. In this state, the decision to end (or not) a simulation is taken, checking if the current $\boldsymbol{p}_1$, $\boldsymbol{p}_2$ and $\boldsymbol{p}_3$ have collapsed into a small triangle.

The outputs of MainGuidance also contain the name of the state that must be activated with the following input. The Navigation model sends this name back, with the last ASV state and the *request* signal to ask for new waypoints. Hence, most of the time, the output will be sent to the Navigation model through the *o_out* port. However, when the next model state is 'loop_start', the output port is *o_reset*, which sends the state name to the Feedback model, and this last model sends it again to the MainGuidance model. Finally, whenever the state 'loop_start' ends, the stored values of the intermediate waypoints; of the current $\boldsymbol{p}_1$, $\boldsymbol{p}_2$, and $\boldsymbol{p}_3$, and all their function values $f(\boldsymbol{p}_k)$ are sent to the Registry model.

3.6 Guidance Model for Determining Contour Curves

This Guidance model for determining contour curves, supported by PAT, is an atomic DEVS model which runs on external transitions, except for an internal transition to become passive. Its behavior is schematized by the states diagram of Fig. 3(b). In the 'initial' state, a predefined waypoint, obtained from a previous study of the water body and placed inside the target contour curve, is sent to the Navigation model to let the ASV reach a point inside that curve. Next, to let the ASV find an external point of the contour curve, the ASV has to move straight forward with no destination by sending a 'fixed_angle' to the Navigator from the 'fixed_angle' state. Next, this Guidance model will alternate between the 'loop' and 'passive' states until the end of the simulation. The 'loop' state always returns a single waypoint. Once the last waypoint is close enough to any of the two initial waypoints, the 'end' state is activated and sends a shutdown message to the Navigation model, which will forward it to its surrounding models.

As happened with the previous Guidance model, the decision on which state to activate is based on the *name* of the corresponding input, established in the last active state of the Guidance model or updated during the

Navigation model operation. Finally, note that this Guidance model only sends data to the Registry model at the end of the simulation instead of at every iteration (as the MainGuidance model does).

4 SIMULATION RESULTS

As mentioned above, this modular model-driven specification is designed to be reusable, scalable, and adaptable to further requirements (e.g., other types of missions or different ASVs). This provides multiple benefits for old and new systems engineers, who may extend or modify only certain parts without knowing all the system's details. A model-driven procedure also fits perfectly with the work distribution inherent to a large development team, where a big project is often divided into smaller pieces of work, such as the one presented in this paper.

In this study, we will test, over the Scenario presented in the following section, the behavior of the complete system for its two different types of missions. The results of the simulations will be summarized in Fig. 4.

4.1 Simulation Scenario

Our ASV explores a variable of interest (e.g., the concentration of cyanobacteria) in a lake region of 1200x1200 m^2. As for simulation purposes, we need to define $f(\cdot)$ to obtain the measurements of the variables of interest at the ASV locations. We will use a function with several optima distributed within the exploration area. It is also worth noting that although different regions of the selected function are displayed as colored curve maps in Fig. 4(a) and 4(b), only the values sampled from the ASV path will be known by the system. Finally, Table 2 summarizes the values of different parameters of the models.

4.2 Seeking Extreme-points of Maximal Concentration

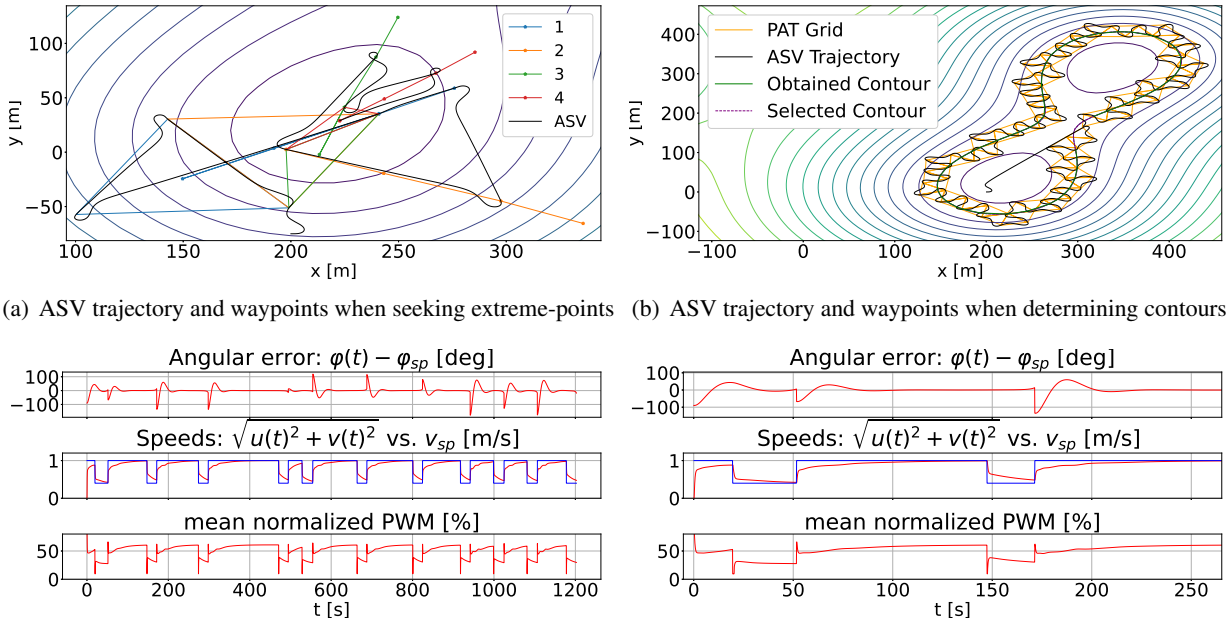
In this mission, the initial ASV state is $\boldsymbol{s}(0) = [200, -75, 0, 0, 0, 0]$, and the starting points of NMSOA are $\boldsymbol{p}_1 = [199, -51]$m, $\boldsymbol{p}_2 = [143, 31]$m and $\boldsymbol{p}_3 = [100, -57]$m. Note that $\boldsymbol{p}(0) = [200, -75]$ is close to one of the extreme points to reduce the number of iterations of NMSOA and to simplify Fig. 4(a). Here, the complete view of the ASV trajectory is represented in black, and the underlying triangles defined by each iteration $\boldsymbol{p}_1$, $\boldsymbol{p}_2$ and $\boldsymbol{p}_3$ are highlighted in a different color (see the legend of the graphic for further information). It is worth noting that the ASV trajectory is not yet smooth enough in the vertex of the triangles, which implies that the parameters of the GNC must be better tuned. Nevertheless, we are postponing this tuning until more data on the real-world ASV is available to fine-tune better the parameters of the ASV model before fine-tuning any further the parameters of the GNC.

Fig. 4(c) shows the temporal evolution of several simulation signals, while Fig. 4(d) shows an amplified view on the first 250 seconds of the simulation. The curve at the top of both figures represents the error (difference) between the actual orientation of the ASV $\phi(t)$ and the course setpoint ϕ_{sp} required by the Navigation model. The sudden changes in these curves show how, whenever a waypoint is reached and a new one becomes available, the ASV has to redirect itself towards the course setpoint ϕ_{sp} associated with the new waypoint. The plateaus of the null value show when the ASV is already well-oriented. The blue

Table 2: Simulation parameters.

Navigation Parameters	*c_period* [s]	d_{speed} [m]	v_{sp}^{high} [m/s]	v_{sp}^{low} [m/s]	d_{reach} [m]
Value	0.1	15	1	0.4	1.5

NMSOA Parameters	ρ	δ	γ	σ	d_{area} [m^2]	PAT Parameters	T [m]	d_{stop} [m]
Value	1	1.5	0.3	0.5	20	Value	30	15



(a) ASV trajectory and waypoints when seeking extreme-points

(b) ASV trajectory and waypoints when determining contours

(c) Evolution of relevant signals when seeking extreme-points

(d) Zoom over the initial 250 seconds of Fig. 4(c).

Figure 4: Simulation results.

curve at the middle of both figures represents the speed setpoint v_{sp} required by the Navigation model, while the red ones represent the ASV speed $\sqrt{u(t)^2+v(t)^2}$. The steps in the blue curves are triggered whenever the ASV has to go to a new waypoint (which makes $v_{sp} = v_{sp}^{high}$) or whenever the ASV location is closer than d_{speed} to the current waypoint (which makes $v_{sp} = v_{sp}^{low}$). The red curves show how the ASV speed arrives at the v_{sp} in an over-damped fashion, implying an unaggressive control of the specialized speed BPI-AW regulator. Finally, the bottom curves of both graphics show the mean of the normalized PWM signals (which are calculated by translating a_i into the range $[-100\%, 100\%]$). These curves have a similar tendency as the speed ones since the main contribution of the changes in the PWM signals corresponds to the requested speed changes. Besides, it is worth noting that their values fall within the $[0\%, 60\%]$ range, which implies that the current tuning of the regulators avoids saturating the PWM signals. Finally, the total duration of the simulation, 1200 seconds = 20 minutes, already falls (without any further fine-tuning) inside the expectations for an ASV monitoring mission of a water body in a real-world scenario.

4.3 Determining a Contour Curve of Constant Concentration

For this simulation, the starting state is $\boldsymbol{s}(0) = [200, 0, 0, 0, 0, 0]$, the point inside the curve is $[200, 20]$m, and the targeted curve level 0.096. The resulting trajectory plot is shown in Fig. 4(b), where the ASV path is represented in black, the PAT triangle grid in orange, the determined contour curve in green, and the selected contour curve (which is unknown by the system) in purple. As both contours overlap, the underlying purple one can only be seen clearly in the area where the PAT algorithm starts working. Moreover, this overlapping confirms that the GNC works correctly, letting the ASV determine curves where the variable of interest remains constant. Finally, it is worth noting that the evolution graph of some variables of interest is not displayed in this case because they have similar overall behavior to those curves in the other mission. The only change between both simulations is the Guidance model.

5 CONCLUSIONS AND FUTURE WORK

This paper presents a DEVS specification for simulating and evaluating the ASV response to a GNC system developed to determine extreme points or contour curves of a variable of interest of a water body. To achieve it, the GNC system is partitioned into several DEVS models, minimizing the dependencies among them to modify some of them in the future to support different types of missions in the Guidance model or try different types of regulators in the Controller. The paper also presents a dynamic model for a catamaran ASV, which is implemented and integrated into the specification using a DEVS atomic model. The paper results show that the specification is working as expected for the two types of already supported missions.

Several improvements are already planned for future work. On the one hand, we will have to finely tune the ASV parameters to make it work as our real-world ones. After this, we will finely tune the GNC parameters to improve ASV trajectories further. Another exciting line of research will be to take advantage of DEVS to simulate multiple ASVs simultaneously and explore the water body more quicker. To achieve it, we will also require to include an ASV-collision avoidance subsystem into the GNC.

ACKNOWLEDGMENTS

This work has been supported by the Research Projects IA-GES-BLOOM-CM (Y2020/TCS-6420) of the Synergic program of the Comunidad Autónoma de Madrid, SMART-BLOOMS (TED2021-130123B-I00) funded by MCIN/AEI/10.13039/501100011033 and the European Union NextGenerationEU/PRTR, and INSERTION (PID2021-127648OB-C33) of the Knowledge Generation Projects program of the Spanish Ministry of Science and Innovation. Ferrero-Losada's work is specifically supported by the "Investigo" Grants of the Comunidad Autónoma de Madrid and the Ministry of Labour and Social Economy of Spain, funded by the Next Generation Program of the European Union.

REFERENCES

Besada-Portas, E., J. M. Girón-Sierra, J. Jiménez, and J. A. López-Orozco. 2021. "Data-Driven Exploration of Lentic Water Bodies with ASVs Guided by Gradient-Free Optimization/Contour Detection Algorithms". In *2021 Winter Simulation Conference (WSC)*, edited by S. Kim, B. Feng, K. Smith, S. Masoud, Z. Zheng, C. Szabo, and M. Loper, pp. 1–12. JW Marriott Desert Ridge, Phoenix, AZ (Virtual).

Bordón-Ruiz, J. B., E. Besada-Portas, J. A. López-Orozco, and J. L. Risco-Martín. 2021. "DEVS-based simulation for search and rescue missions involving multipleUAVS". In *Annual Modeling and Simulation Conference (ANNSIM)*, edited by C. R. Martin, M. J. Blas, and A. I. Psijas, pp. 1–12. Fairfax, VA, USA.

Carazo-Barbero, G., E. Besada-Portas, J. M. Girón-Sierra, and J. A. López-Orozco. 2021. "EA-Based ASV Trajectory Planner for Pollution Detection in Lentic Waters". In *Applications of Evolutionary Computation*, edited by P. A. Castillo and J. L. Jiménez Laredo, pp. 812–827. Cham, Springer International Publishing.

European Comission 2023. "European Commission water related directives". https://ec.europa.eu/environment/water/index_en.htm. Accessed on January 2023.

Giron-Sierra, J. M., and J. Chacon-Sombria. 2021. "Application of Teams of USVs for Cyanobacteria Monitoring: Initial Steps". In *3th IFAC Conference on Control Applications in Marine Systems, Robotics, and Vehicles (CAMS)*, edited by A. Hahn, pp. 812–827. Oldenburg, Germany.

Hitz, G., F. Pomerleau, M. Ève Garneau, C. Pradalier, T. Posch, J. Pernthaler, and R. Siegwart. 2012. "Design and Application of a Surface Vessel for Autonomous Inland Water Monitoring". *IEEE Robotics & Automation Magazine* vol. 19, pp. 62–72.

Holman, K., J. Kuzub, and G. A. Wainer. 2010, April. "UAV Search Strategies Using Cell-DEVS". In *Proceedings of 2010 Spring Simulation Conference (SpringSim10), ANSS Symposium*, edited by S. Biaz, E. Imsand, and S. Wu, pp. 192–199. Orlando, FL, SCS.

Liu, Z., Y. Zhang, X. Yu, and C. Yuan. 2016. "Unmanned surface vehicles: An overview of developments and challenges". *Annual Reviews in Control* vol. 41, pp. 71–93.

Mezher, D., and B. Philippe. 2000. *PAT-a Reliable Path Following Algorithm*. Rennes, France, INRIA. Technical report.

Moreno, A., L. de la Torre, J. L. Risco-Martín, E. Besada-Portas, and J. Aranda. 2011. "DEVS-based Validation of Uav Path Planning in Hostile Environments". In *Proceedings of the 2011 International Mediterranean & Latin American Modelling Multiconference (IMMM 2011)*, edited by A. Bruzzone, W. Buck, and J. A. Sokolowski, pp. 1–6. Rome, Italy.

Nelder, J. A., and R. Mead. 1965, 01. "A Simplex Method for Function Minimization". *The Computer Journal* vol. 7 (4), pp. 308–313.

Risco-Martín, J. L., S. Mittal, K. Henares, R. Cardenas, and P. Arroba. 2022. "xDEVS: A toolkit for interoperable modeling and simulation of formal discrete event systems". *Software: Practice and Experience*, pp. 1–42.

Shuo, J., Z. Yonghui, R. Wen, and T. Kebin. 2017. "The Unmanned Autonomous Cruise Ship for Water Quality Monitoring and Sampling". In *2017 International Conference on Computer Systems, Electronics and Control (ICCSEC)*, pp. 700–703. Dalian, China.

Siyang, S., and T. Kerdcharoen. 2016. "Development of unmanned surface vehicle for smart water quality inspector". In *2016 13th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, pp. 756–761. Chiang Mai, Thailand.

United Nations 2023. "2030 Agenda for Sustainable Development". https://sdgs.un.org/goals. Accessed on January 2023.

United States Environmental Protection Agency 2023. "Drinking Water Requirements for States and Public Water Systems". https://www.epa.gov/dwreginfo/drinking-water-regulations. Accessed on January 2023.

Zeigler, B. 2019. *Theory of modeling and simulation : discrete event and iterative system computational foundations*. London, United Kingdom, Academic Press, an imprint of Elsevier.

AUTHOR BIOGRAPHIES

SAMUEL FERRERO-LOSADA is a junior researcher at the University Complutense of Madrid (UCM). He holds a Master degree in Astrophysics from the same University. His research interests lie in modeling and simulation. His email address is saferr03@ucm.es.

EVA BESADA-PORTAS is an Associate Professor of Systems Engineering and Automation at UCM. She also holds a PhD in Computer Systems from UCM. Her research interests include uncertainty modeling and simulation, optimal control and planning of unmanned vehicles. Her email address is ebesada@ucm.es.

JOSÉ L. RISCO-MARTÍN received his Ph.D. from UCM, where he currently is Full Professor in the Department of Computer Architecture and Automation. His research interests include systems modeling, simulation, and optimization. His email address is jlrisco@ucm.es.

JOSÉ A. LÓPEZ-OROZCO is a Full Professor in the UCM. He holds a Ph.D. in Physics from the same University. His research interests include multisensor data fusion, control and planning of unmanned vehicles, and robotics. His email address is jalo@ucm.es.