DEVS-based evaluation of UAVs target-search strategies in realistically-modeled missions

XXXXXXX
XXXXXXX
XXXXXXX
XXXXXXX
XXXXX
XXXXX
XXXXX
XXXXX
XXXXXXXXX XXXXXXX XXXXXXX XXXXXXX XXXXXXX XXXXX XXXXXX
XXXXXXX, XXXXXXX

ABSTRACT

Searching for targets from a group of Unmanned Aerial Vehicles (UAVs) is a complex problem whose applications range from the localization of military targets to search and rescue missions. Determining the best locations to search within the mission scenario requires to consider the dynamics of the UAVs and of its onboard sensors, and the uncertainty of the problem, usually related with the target initial location and dynamics, and with the sensor likelihood. Besides, what is best is not always the same (e.g. it can be maximizing the detection probability and/or minimizing the target detection time, while ensuring communications, smooth trajectories, energy saving, etc). These makes the evaluation of UAVs target-search strategies a complex system itself. In this paper, we tackle this problem using the Discrete Event System Specification (DEVS) to exploit its modular and hierarchical design, and to improve the reusability and scalability of our evaluation system. DEVS also provides simple and clear semantics to manage the complexities of the system, represents an explicit separation between the model specification and the corresponding simulation, and helps us to debug and verify our model, as the results of the paper show.

CCS CONCEPTS

• **Computing methodologies** → **Modeling and simulation**; *Probabilistic reasoning*; *Planning under uncertainty*.

KEYWORDS

Discrete Event Systems, Model Based Systems Engineering, Bayesian Search, Multi-Objective Path Planning, Multi-Agent Systems

ACM Reference Format:
XXXXXXX, XXXXXXX, XXXXXXX, and XXXXXXX. 2021. DEVS-based evaluation of UAVs target-search strategies in realistically-modeled missions. In *Suffolk '21: ACM SIGSIM Conference on Principles of Advanced Discrete Simulation, May 31–June 2, 2021, Suffolk, VA.* ACM, New York, NY, USA, 11 pages. https://doi.org/XXXX.XXXX/XXXXXX.XXXXXX

Suffolk '21, May 31–June 2, 2021, Suffolk, VA

ACM ISBN 978-1-XXXX-XXXX-X/18/06…$15.00
https://doi.org/XXXX.XXXX/XXXXXX.XXXXXX

1 INTRODUCTION

Looking for targets placed at unknown positions within a given area by means of mobile sensors is a problem that has been long studied, since the Second World War [33] and from more recent perspectives that have mainly emerged from the fields of *Operational Research*, *Optimal Control* and *Information Fusion*. Within the first field, the problem is formulated as a Partially Observable Markov Decision Process, where the target location is the unknown state and the information provided by the mobile imperfect sensors are the observations [16, 32]. Within the second domain, it is considered a stochastic optimal control problem, where the sensors displacements are regulated taking into account the information gained by sensing [6, 7]. Finally, formulations within the third discipline exploit Bayes theory to: 1) represent and update the available information of the target state; and to 2) determine the sensors' searching strategy by optimizing probabilistic-based utility functions [2, 9]. The work in this paper is related to this last group, as it is focused on the development of a scalable and reusable implementation of the utility functions required to evaluate target-search strategies that are performed by sensors placed on board Unmanned Aerial Vehicles (UAVs) during real-world missions.

Moreover, the theory and more traditional algorithms that have been developed within the different fields can be useful for numerous search applications (including those involving UAVs as mobile sensing platforms[1]), such as military target detection, search for survivals after natural disasters, or maritime search and rescue missions [8, 18]. Besides, to apply them to real-world target-searches and bring other aspects of the mission into consideration (e.g. non-flying zones, communication losses, UAV dynamics and/or energy savings), new approaches are continuously developed, as those presented and/or reviewed in [15, 21, 26, 28].

Increasing the target-search realism within the mission planner often involves expanding the complexity of the evaluation process required by the optimization algorithms to determine which UAVs

[1] UAVs are often selected due to their capability to overfly and search wide regions in an assumable time span.

search-strategies are the best. In particular, the evaluation process should consider, at least, the UAVs dynamics[2], the uncertainty of the target location and movements, and the sensor likelihood. Besides, depending on the final objectives of the search mission, the evaluation process calculates different utility functions related with the previous elements, such as the probability of detecting the targets [2, 19] or the expected time of detection (to increment the changes of locating them as soon as possible [25, 29]). Additional objectives, related to other mission aspects (e.g. those previously mentioned), can also be needed. In the end, the evaluation process can become a complex task, with a big set of models interacting. Besides, these models can be simulated at different levels of resolution, depending on the precision defined by the operator before each simulation. Using Model Based System Enginering (MBSE) and Discrete Events Simulation (DES) principles can help us to define and manage this complex task. On one hand, MBSE allows a clear separation between models specification and the simulation process, facilitating the incremental models design and refinement. On the other, DES supports multi-resolution modeling, as well as the integration of different types of models (e.g. probabilistic and deterministic, continuous and discrete, etc.), which can be evaluated at different rates or at asynchronous instants. Both paradigms also help us to debug, verify, and validate this complex evaluation process.

This paper presents our approach to systematize the evaluation process of UAV target-search strategies using MBSE and DES. In particular, we use the Discrete Event System Specification (DEVS, [42]), a well-established DES formalism that is gaining acceptance with a holistic construct called the Modeling and Simulation Framework. Additionally, DEVS has been successfully used in other complex systems (such as, to name a few, [10, 12, 23, 30]) in order to take advantage of its capabilities to conduct the MBSE process and of its versatility to handle multiple resolution modeling levels [41].

To implement and analyze the new approach, we select the evaluation process required by the UAV trajectory planners presented in [25, 26], as they already incorporate complex models for the UAV dynamics and sensors, and evaluate multiple utility functions. The paper also analyzes, through several examples, the possibilities that the new DEVS-based evaluation system brings for future optimizations of target-search strategies and the effects of the resolutions used during the simulation of the models in the final values obtained for the evaluation criteria.

2 RELATED WORKS

On one hand, this section discusses different approaches and formulations of the target-search problem. It is not an exhaustive state of the art, as it focuses on highlighting the main characteristic of different types of target-search mission planners based on Bayes theory and on introducing the evaluation approach of those works that have motivated this paper. During this discussion, we also present the advantages of bringing more flexibility to the evaluation process, to be able to develop in the future more versatile planners. On the other one, we also review a few works that use DEVS for closely related problems that involve UAVs or planning.

2.1 Target-Search Related Works

Let us analyze the selected target-search publications from the perspective of the elements and models of their evaluation process.

2.1.1 Targets uncertain location and displacement. They are usually modeled under a probabilistic umbrella, that in some cases capture the chances that the target is at each location/region of the search space [1, 3–5, 19–21, 25, 26, 29, 34, 37, 40] and in others how probable is to locate targets in a given region [14, 39, 43][3]. Besides, within the first group (where the selected works to implement our approach fall), the probability distribution is often modeled as a probability map over a grid of cells or as a weighted sum of Dirac functions centered at different samples of the search space, and updated (accordingly to the target movement and sensory uncertainties) with Recursive Bayesian Filters (BFs, [4, 5, 20, 25–27, 34, 37, 38, 40]) or Particle Filters (PFs, [1, 3, 19]).

As BFs and PFs share the steps used to update their corresponding probability distributions but differ in their properties and in the calculations that implement them, it is undoubtedly useful to be able to substitute one for the other. However, only one planner considers both possibilities [29], in spite of the advantages associated with each probability model and filter.

2.1.2 Sensors observing the search space. While looking for targets, sensors take measurements from the positions allowed by the movements of the UAVs in which they are embarked. However, their observations have a limited range and can fail. This behavior is modeled through likelihood functions, which are used by the BFs and PFs to update the probability distributions. Their expressions depend on the type of sensors and on the realism of the planner, ranging from ideal/constant sensor models that only observe a few cells of the probability map under the UAV location [5, 21, 27, 34, 38, 40], to range-based sensor models with probability curves that decrease with the distance between the sensor and target location [19, 20, 37], or to specific models of certain types of radars [25] and cameras [4, 26, 29].

Again, being able to easily change the sensor likelihood is useful to re-use the planner for UAVs equipped with different sensors. Although the previous works usually present the results related to a specific sensor type, the majority could be easily applied to others. However, it could also be useful to 1) move the sensor orientation and location, or 2) decide when and how often to activate them. Among the reviewed works, the first option is considered only in [26, 29], while the second one is not yet included.

2.1.3 UAVs trajectories. They are defined/encoded by the planners with different strategies and according to different assumptions. On one hand, several works [1, 17, 19] obtain trajectories traditionally used in search missions by adjusting the parameters (e.g. initial location, length, width) of predefined patterns (e.g. lawn-mower and spiral). On the other one, and specifically in works whose underlying target model is a probability map, the trajectories are defined as straight lines that joint the center of a cell with the center of another [5, 21, 27, 29]. Finally, a third group of planners that manipulate the periodical setpoints of the UAVs (e.g. heading,

[2]Including the dynamics of its onboard sensors if they also move/reorient themselves from its attachment to the UAVs.

[3]Both cases are different because in the first the probability function is distributed over the search space and in the second there is a probability function over each region.

speed and height) and exploit their dynamical models to obtain free-shape trajectories [4, 20, 25, 26, 34, 37, 40][4].

Using dynamical models to generate the UAVs trajectories ensures the fulfillment of the maneuverability constraints of different types of UAVs (e.g. fixed vs. rotatory wing ones) and allows to include environmental aspects (e.g. winds) into the evaluation processes of the planners. In the two other cases, although the trajectories are easier to understand by human pilots, their feasibility should be checked (or ensured by construction). Again, the capability of changing how the evaluation process obtains the UAVs trajectories from their planner encoding can be useful to re-use other elements of the evaluation process. This property is not supported by the works under analysis, what also makes the comparison of target-search strategies a troublesome task.

2.1.4 Utility functions. They exploit the information provided by the models associated with the previous elements to evaluate a given target-search strategy. They can be classified in two groups:

(1) *Probability-based utility functions,* which refer to those objectives that relate the (targets & sensors) probability models and filtering processes with the (UAVs & sensors) trajectories. Often, they are the probability of detecting the target from the sensor trajectory [4, 5, 19–21, 34, 37, 38, 40] and the expected time to detect the target [25–27, 29]. The works under analysis usually focus on optimizing one of those functions, although it can be useful to simultaneously evaluate or optimize them, or substitute one for another promptly.
(2) *Mission-specific utility functions,* which refer to the remaining objective and constraint criteria evaluated and optimized by the planners. Although these functions bring realism and additional intends to the missions, just a few works under analysis [25–27] consider these types of functions, and specifically evaluate: if UAVs pass over nonflying zones, if there are collisions between them, if they are able to maintain a communication network with the ground control station or their fuel consumption.

2.1.5 Numbers of targets, UAVs and sensors on board each UAV. These are other variables to consider, as they usually increment the computation requirements and complexity of the evaluation process. In particular, the evaluation process of the works under analysis varies from single-target [1, 4, 5, 25–27, 29, 34, 38, 40] to multi-target [20, 21, 37], and from single-UAV [1, 5, 38] to multi-UAV [4, 20, 21, 25, 26, 29, 34, 37, 40]. Besides, they only consider a single target-detection sensor within each UAV.

Facilitating the change in the number of all these elements as well as allowing to modify the underlying models of the different instances is interesting to build planners that simultaneously exploit the capability of different types of UAVs equipped with multiple sensors and that consider the situation of multiple targets.

2.1.6 Final Remarks. The previous analysis shows that we can improve the evaluation process of existing planners by combining their possibilities or including new capabilities. To achieve it, it can be extremely useful to implement the evaluation process under a flexible well-established Model and Simulation methodology (like MBSE and DES through the use of DEVS) that facilitates the integration of different types of models (for the UAVs, sensors, and targets) at different resolutions, as well as the adaptation to different types of target-search missions. Nevertheless, to the best of the authors' knowledge, the existing works do not focus on this aspect of the research, which will undeniably allow us to improve the management of these models, and to provide support for verification, validation, as well as reliability and scalability analysis.

2.2 DEVS related works

Next, we present several works that apply DEVS formalism to closely related problems, involving UAVs and/or planning.

The closest work is [13], as it presents a planner for determining the trajectory of a UAV that wants to maximize the probability of detecting a target. To do it, it uses Cell-DEVS[5] to model the target-search problem with cellular automatas that combine diffusion rules to update the probability map and high-climbing algorithms to determine the UAV search-pattern. Our approach differs from [13], as our atomic and coupled models are associated with the different elements of the search problem, include more realistic behaviors (e.g. UAV, sensor, and target dynamics), and our system focuses, so far, in the evaluation process of given UAVs trajectories. Besides, following the Cell-DEVS formalism, other path, defense or emergency planners/simulators are presented/surveyed in [35, 36].

Besides, and in chronological order, the following works involving DEVS and UAVs have appeared. On one hand, [10] presents Pliades, a DEVS based simulator developed to systematically and intensively evaluate the effectiveness of military missions involving multiple vehicles. Although our evaluation system is originally intended to be part of a planner, it can also be used to run Monte-Carlo simulations of target-search scenarios, providing an extra atomic model that randomly modifies some of the inputs of our evaluation system. On another one, [23] presents a DEVS-based model to evaluate the trajectories returned by a planner for UAVs that have to overfly an ordered list of way-points avoiding radars, missiles, and non-flying zones. As the elements in that problem are different from the ones in ours, the atomic and coupled models of both works are not the same, although the way of developing both evaluation systems is rather similar. Finally, [24] presents a unified DEVS-based platform to model and simulate hybrid control systems, which is validated over a system intended to deal with mapping missions involving a fixed-wing UAV. The system includes multiple models for the UAV dynamics and controller, and a streamlined motion and path planner, which makes the UAV follow pre-defined patterns (e.g. lawnmower) over the mapping region. Although this work is more focused than ours in the UAV lower-level and path-following simulation, some of their ideas can be included in ours to simulate the UAV trajectory using a different model decomposition than the one that is currently implemented in our system.

The previous analysis shows how DEVS has already been successfully used to model and simulate systems involving UAVs, performing different types of missions. They also suggest that it can also be an interesting tool to develop a framework for evaluating (and in the future for planning) target-search UAV-strategies.

[4]Within this group, note that [4, 20, 34, 37, 40] use a streamlined differential model, while [25, 26] use a more complex one that includes height, speed and lateral dynamics.

[5]Cell-DEVs is a DEVS-based model specification for cellular automatas with explicit timing delays.

3 BACKGROUND

This section describes the evaluation process of the planners in [25, 26] and the main characteristics of DEVS, all of them selected to implement and analyze the approach presented in this paper.

3.1 Formulation of the Search Problem

This section summarizes the main variables, properties and models selected to verify and validate our new DEVS-based framework to evaluate target-search strategies for UAVs[6].

3.1.1 Target-related operations. The rectangular search region Ω, parallel to the (x, y) axes for simplicity, is discretized into a Grid of $N_G = N_x$ x N_y rectangular cells, each of size w_x x w_y.

The discretization of Ω is required, since the target probability distribution $b(\tau^t)$ at a given time t is discretized as a probability map - mass function - $b(c^t)$ over each cell $c^t \in G$. The initial target belief $b(\tau^0)$ is also discretized as $b(c^0)$, and updated to obtain $b(c^t)$ using the following operations related to the steps of the RBF:

Initialization, making the unobserved probability $p(c^0)$ over the cells $c^0 \in G$ equal to $b(c^0)$.

Prediction, which redistributes the unobserved probability over the map as the target moves. To do it, the operation stated at Eq. (1) is carried out, exploiting the target motion model $p(c^t|c^{t-T_\tau})$, which expresses how probable is that the target at cell c^{t-T_τ} at time step $t - T_\tau$ arrives at cell c^t at time t. As the target motion model is usually defined for a fixed time lapse T_τ, this step is usually applied periodically.

$$p(c^t) \leftarrow \sum_{c^{t-T} \in G} p(c^t|c^{t-T_\tau}) p(c^{t-T_\tau}) \tag{1}$$

Assimilation, which updates the unobserved probability over the map with the information provided by the sensors on board the UAVs. This operation is performed with Eq. (2) whenever UAV u has to take a measurement with its k-th sensor to either detect D or not detect $\overline{D}$ the target. To do it, it is necessary to know the probability of not detecting the target $p(\overline{D}|c^t, s^{t_m}_{u,k})$ placed at cell c^t from the location and pose $s^{t_m}_{u,k}$ of that sensor at time stamp $t_m \in [t, t + T_\tau]$[7].

$$p(c^t) \leftarrow p(\overline{D}|c^t, s^{t_m}_{u,k}) \cdot p(c^t) \tag{2}$$

Besides, $p(\overline{D}|c^t, s^t_{u,k})$ is obtainable from the sensor likelihood $p(D|\tau^t, s^{t_m}_{u,k})$, which states how likely is measuring τ^t from $s^{t_m}_{u,k}$. The easiest way, performed in [25, 26], is $p(\overline{D}|c^t, s^t_{u,k}) = 1 - p(D|\tau^t, s^t_{u,k})$ considering that τ^t is the center of c^t. However, if the likelihood changes significantly within the cell, it is better to calculate the mean of the likelihood over a set of N_c equally spaced points within c^t (i.e. $p(\overline{D}|c^t, s^t_{u,k}) = 1 - \sum_{\tau^t \in points(c^t, N_c)} p(D|\tau^t, s^t_{u,k})/N_c$).

[6]Further details can be found in the original papers [25, 26].

[7]Note that Eq. (2) allows to include several measurements taken at same t_m and assimilates all the measurements between two target predictions to the time step of the first. Moreover, those readers familiar with RBF may have noted that Eq. (2) lacks of normalization term and that only considers the non-detection measurement. This happens because Eqs. (1) and (2) are finally intended to estimate the expected time of detection [25, 26]. Nevertheless, obtaining the target belief $b(c^t)$ only requires calculating $p(c^t)/\sum_{g \in G} p(c^t = g)$ when desired.

Finally, we summarize the models of this section that are inputs of the evaluation process: $b(c^0)$, $p(c^t|c^{t-T_\tau})$ and $p(D|\tau^t, s^{t_m}_{u,k})$.

3.1.2 UAV-related operations. The UAV and sensor trajectories used in [25, 26] are obtained from a non-linear dynamical model implemented in Simulink [31], whose inputs are the initial state s^0_u of the UAV and of its sensors $s^0_{u,k}$, and a sequence of setpoints for the UAV heading, height and speed, and for the sensor pose. The model in [25] includes the fuel, height, speed and lateral dynamics of the UAV, while the model in [26] incorporates the pose dynamics of a gimballed camera. Both models include the usual limitations related with their state variables (e.g. air velocity, height, heading, sensor pose), as well as their dependencies with the wind.

As the differential equations of these models do not bring much insight to the discussion of this paper, we summarize them with the following expressions. Besides, to smooth the comparison of the DEVS-based UAV trajectories with those obtained with Simulink, we have implement them as a group of difference equations[8]:

- $s^t_u = f(s^{t-T_u}_u, a^{t-T_u}_u, \epsilon^{t-T_u}, T_u)$ stands for the expressions used to compute the new state s^t_u of UAV u from its previous state $s^{t-T_u}_u$ given the set points in $a^{t-T_u}_u$, the environmental conditions (e.g. wind) in ϵ^{t-T_u}, and the sampling period T_u.
- $s^t_{u,k} = g(s^t_u, s^{t-T_{u,k}}_{u,k}, a^{t-T_{u,k}}_{u,k}, \epsilon^{t-T_{u,k}}, T_{u,k})$ stands for the expressions used to compute the new state $s^t_{u,k}$ of sensor k of UAV u from the current UAV state s^t_u and the previous sensor state $s^{t-T_{u,k}}_{u,k}$, given the sensor setpoints in $a^{t-T_{u,k}}_{u,k}$, and $\epsilon^{t-T_{u,k}}$ and $T_{u,k}$.

Finally, it is worth noting that in [25, 26], both models iterate at the same basic time step (i.e. $T_u = T_{u,k}$) and only accept changes in all setpoints at the same fixed rate, pre-defined before hand. This restriction will be alleviated in this paper.

3.1.3 Evaluation-related operations. The models and equations in the previous sections do not yet determine the utility of a given set of UAV trajectories (defined through the setpoints of the UAVs and of the sensors). This section introduces those functions:

- The probability-based utility function are the probability of detection up to t ($P_d(t)$) and the expected time of detection ($ET(t)$), which are computed with Eqs. (3) and (4).

$$P_d(t) = 1 - \sum_{c^t \in G} p(c^t) \tag{3}$$

$$ET(t) = \sum_{l=1:1:t/T_\tau} (1 - P_d(l \cdot T_\tau)) T_\tau \tag{4}$$

- The remaining utility functions are:
  - Fuel consumption, computed by the UAV motion model.
  - Collision avoidance, which determines how often each pair of UAVs is closer than the security flight range.
  - Non flying zones (NFZs), which determines how often the UAVs overfly forbidden regions, which are defined as a list of non-flyable cells $c^{NFZ}_l \in G$.

[8]These equations itself integrate, using a 4th order Runge-Kutta, the differential equations of the models.

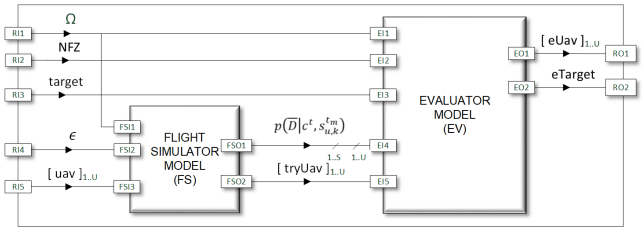


Figure 1: Root coupled model

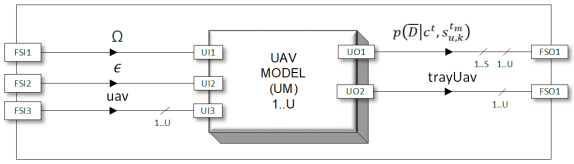


Figure 2: Flight simulator (FS) coupled model

3.2 DEVS Formalism

As stated above, our methodology to define our new framework for evaluating UAVs target-search strategies is based on MBSE and DES principles. To this end, we have selected the DEVS formalism [42]. As a result, we provide a common architecture with a highly-detailed structural and behavior description, and use DEVS to validate and guarantee an incremental, reliable and solid design, with an explicit separation from the chosen implementation. These characteristics facilitate scalability, maintainability, and reusability. Additionally, introducing new system elements (i.e. a new type of UAV or of sensor) is easier and faster, as we will see below. Trade-offs between new or existing elements can also be performed, in order to analyze which element fits better in a specific mission.

Overall, DEVS enables to represent a system by three sets (input X, output Y and state S) and five functions (external transition δ_{ext}, internal transition δ_{int}, confluent δ_{con}, output λ, and time advance ta). Besides, DEVS models are either atomic and coupled. On the one hand, atomic models define the system's behavior through the five functions mentioned above. They process input events based on their model's current state and condition, generate output events and transitions to the next state. On the other one, coupled models are the aggregation/composition of several atomic or coupled models connected by explicit couplings. Given this recursive definition, a coupled model can be a component of a larger coupled model system. Hence, DEVS supports a hierarchical model construction that is exploited to define our framework.

4 DEVS-BASED EVALUATION PROCESS

This section presents the specification for our DEVS-based target-search evaluation tool. Designed using MBSE principles and exploiting DEVS scalability, reusability and flexibility, it presents a modular design capable of adapting to different types of scenarios. Moreover, depending on the particular characteristics of each element used in a given mission (e.g. UAVs can be equipped with static radars or gimballed cameras), our models are reconfigured to match the actual behaviors of each element. Next, we describe the structure and behavior of each model of our evaluation framework.

4.1 Root Model

The Root Model is the top-level coupled module of the architecture and represents our DEVS-based evaluation tool for the search problem. As Fig. 1 shows, it is composed by the Flight Simulator (FS) and Evaluator (EV) coupled models, which are used to obtain the UAVs (and sensors) trajectories while the target evolution (due to its own movement and to the observations from the UAVs) and utility functions are evaluated. More in detail, the ports of this model are:

(1) The search area Ω, defined by its geographical coordinate origin and its lateral dimensions.
(2) The NFZs, defined as the lists of cells of $c_l^{NFZ} \in G$.
(3) The encapsulated definition of the `target`, which includes its initial belief $b(c^0)$, its motion model $p(c^t|c^{t-T_\tau})$, and its end time t_τ^{end}.
(4) The information of the environment ϵ, which so far consists of a wind matrix of the same size as G.
(5) The encapsulated definition of the UAVs involved in the mission $[\texttt{uav}]_{1:U}$, which for each UAV includes its initial state $s_u^{t_u^0}$, its initial and final time in the mission $[t_u^0, t_u^{end}]$, its deterministic motion model $f(\cdot)$, the list of UAV setpoints (i.e. as many a_u^t as desired) , and the information of its onboard sensors, which is described later.

This model's outputs reflect the results of the evaluation process:

(1) The encapsulated definition of the UAVs simulations $[\texttt{eUav}]_{1:U}$, which for each UAV includes the resulting UAV trajectory for the given setpoints and the values of the utility-functions UAV collisions, NFZs overflight and fuel consumption.
(2) The encapsulated definition of the target evaluation `eTarget`, which include the evolution (for different values of t) of the unobserved probability $p(c^t)$, and of the values of the utility-based functions $P_d(t)$ and $ET(t)$.

4.2 Flight Simulator (FS)

The FS coupled model is where the simulation of each UAV trajectory takes place. Consequently, and as Fig. 2 shows, the FS model creates for each UAV u in the mission one instance of the UAV Model (UM), whose structure and behavior is presented below.

4.3 UAV Model (UM)

Each UAV u is represented by an instance of the UAV coupled model, which is composed (as Fig. 3 shows) by the UAV Control (UC) and UAV Motion (UMM) atomic models, and by as many Sensor coupled Models (SM) as target-detection sensors are mounted in the UAV. Besides, the UM model receives as input data the search area Ω, the `environment` information and the `uav` definition. Finally, its output ports are $p(\overline{D}|c^t, s^{t_m}_{u,k})$ for each onboard sensor at its given measurements times t_m, and the simulated trajectory `trayUAV`.

4.3.1 UAV Control (UC). The main function of this atomic model is regulating the simulation of the UAV trajectory. So, it handles the lists of UAV setpoints (UAV headings, heights and speeds), which are built before the simulation, programmable as periodic or event-based signals, and definable as absolute/incremental/rate setpoints.

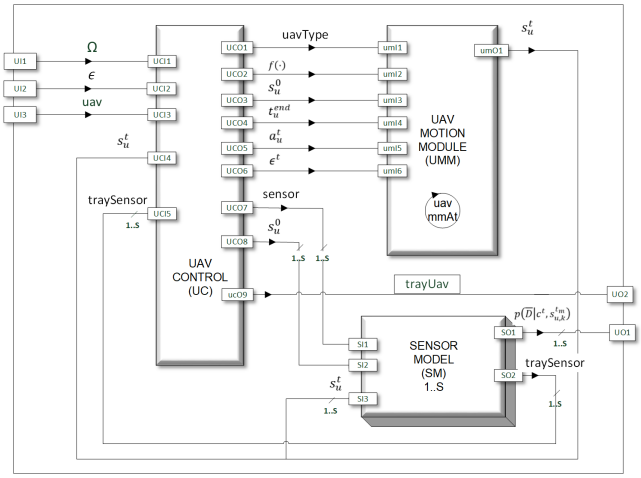


Figure 3: UAV (UM) coupled model

Its behavior is summarized in Fig. 4 and Table 1. In more detail, as soon as this model receives the initial input data (Ω, ϵ, uav), it starts to operate by programming the internal transition δ_{int} and setting σ to the UAV start time t_u^0. By doing so, scenarios with multiple UAVs that have different start mission times can be simulated. Next, at t_u^0, its λ output function sends the required initial data to the UMM and SM models (to allow them to start their own operations).

To handle the setpoints lists, UC programs a δ_{int} by setting σ to the time of the next setpoint. When this time comes, the setpoint is sent by the λ function to the UMM model. This behavior is repeated until the setpoints lists are empty. Besides, a δ_{ext} is triggered each time a new UAV state s_u^t is received through the input port linked to s_u^t output port of the UMM model. When this δ_{ext} is triggered, s_u^t is stored in the UAV path internal list and used to check if new environmental information (in our case the wind speed and direction around the UAV location) has to be sent to UMM model by programming an instant δ_{int} and returning to the previous state. Similarly, sensor simulated trajectories `traySen` can also be received via the input port linked to the SM model, triggering[9] a δ_{ext} that stores `traySen` into the internal sensor list.

When the setpoints lists is empty, UC transitions to a waiting state until the UAV end time t_u^{end} is reached. During this phase, UMM s_u^t or SM `traySen` are received and processed as already described. Finally, when t_u^{end} is reached, an additional δ_{int} is programmed before UC outputs the simulation data and changes to a passive state. This allows UAV dynamic sensors to report their respective simulations when the sensor end time $t_{u,k}^{end}$ is equal to the t_u^{end}. Otherwise, UC goes to a passive state before the SM models report the sensor simulation data.

4.3.2 UAV Motion (UMM). This atomic model represents the UAV flight dynamics. It remains inactive until at time t_u^0, when it receives the initial data from UM. Specific parameters that define each type of UAV are loaded at this moment to configure the behavior of the motion equation $f(\cdot)$, which is executed periodically by UMM. This

[9] As traySen are only sent by SM models of mobile sensors, this behavior is not always triggered.

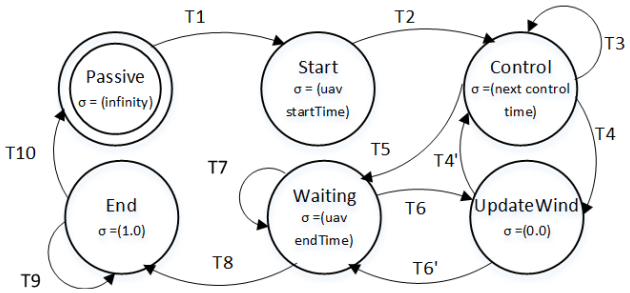


Figure 4: UAV control (UC) states diagram

Table 1: UC States Transitions

	δ_{int}	δ_{ext}
T1		At receiving initial data
T2	At reaching t_u^0	
T3	At next setpoint time	At receiving s_u^t or `traySen`
T4	When new ϵ^t is available	
T4'		To send ϵ^t
T5	When setpoints list is empty	
T6	When new ϵ^t is available	
T6'		To send ϵ^t
T7		At receiving s_u^t or `traySen`
T8	At reaching t_u^{end}	
T9		At receiving `traySen`
T10	end	

is achieved by programming a δ_{int} that runs $f(\cdot)$ and setting σ to the UAV motion rate. Besides, UAV setpoints a_u^t and environmental information ϵ^t are received via their input ports, triggering a δ_{ext} transition every time they are updated by the UC model. When this happens, UMM updates the a_u^t and δ_{ext} that are applied to $f(\cdot)$ until receiving a new input. Finally, note that every time $f(\cdot)$ is executed, a new s_u^t is output to the UC model via the λ function, a behavior that is repeated until this model time exceeds t_u^{end}.

4.4 Sensor Model (SM)

Each target-detection sensor k on board of the UAV has a one to one match to a Sensor coupled model, which in general contains, as Fig. 5 shows, the Sensor Control (SC), Sensor Motion (SMM) and Sensor Payload (SP) atomic models. However, their final configuration depends on the type of sensor: static sensors only require to execute SP, while dynamic sensors require all the couplings and models in Fig. 5. The input ports of these models are the `Sensor` definition, the UAV start time t_u^0 and s_u^t produced by UMM model. In particular, the `Sensor` definition consists in the initial and final time of the sensor in the mission $[t_{u,k}^0, t_{u,k}^{end}]$, its measurement rate t_m and, for mobile sensors, its initial state $s_{u,k}^0$, its deterministic motion model $g(\cdot)$ and the list of sensor setpoints (i.e. as many $a_{u,k}^t$ as desired). Finally, its output ports are $p(\overline{D}|c^t, s_{u,k}^{t_m})$ and the simulated trajectory `traySen`[10].

[10] traySen port is only used by SM models of mobile sensors.

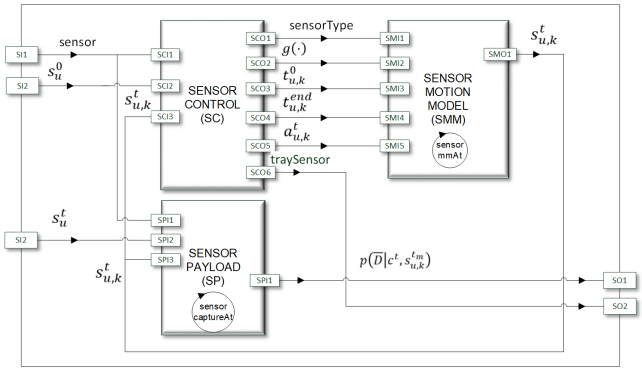


Figure 5: Sensor (SM) coupled model

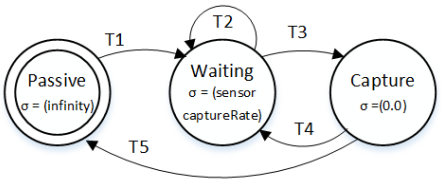


Figure 6: Sensor payload (SP) states diagram

Table 2: Sensor payload (SP) states transitions

	δ_{int}	δ_{ext}
T1		At receiving initial data
T2		At receiving s_u^t or $s_{u,k}^t$
T3	At every sensor t_m	
T4	To send $p(\overline{D}\|c^t, s_{u,k}^{t_m})$	
T5	If next $t_m > t_{u,k}^{end}$	

4.4.1 Sensor Control (SC). The main function of this atomic model is simulating the sensor trajectory by regulating the list of sensor setpoints, whose information is sensor-dependent[11]. As conceptually, it has a similar utility to the UAV control (UC) model, SC behavior is implemented similarly to parts of UC behavior[12].

4.4.2 Sensor Motion (SMM). This atomic model represents the sensor dynamics. Conceptually, as it has an equivalent role for the sensor as the UAV motion model (UMM) for the UAV, their state diagram are similar, differing only in the motion model (i.e. $g(\cdot)$ vs. $f(\cdot)$), and in the input/output signals (related to sensors vs. UAVs).

4.4.3 Sensor Payload (SP). This atomic model is in charge of calculating the sensor likelihood for the target belief, according to the behavior presented in Fig. 6 and Table 2. When it receives its initial information at t_u^0, SP loads the parameters associated to the specific $p(D|\tau^t, s_{u,k}^t)$ implemented for the selected type of sensor and transitions to a waiting state until the sensor start time $t_{u,k}^0$ is reached. Afterwards, it programs a δ_{int} by setting σ to the defined sensor

[11]For instance, in our gimballed camera is its azimuth and elevation, can be represented with absolute/incremental/rate signals and updated periodically or acyclically.
[12]In particular, it does not require that part associated to the environment and to the reception of parts of the sensor trajectory.

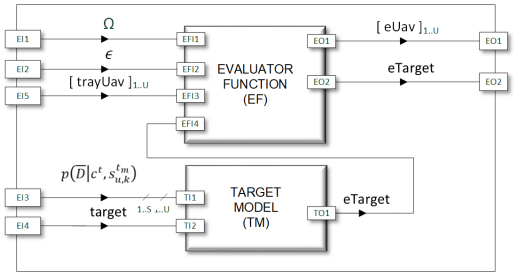


Figure 7: Evaluator coupled Model

measurement rate. When the measuring time t_m is reached, its λ function outputs $P(\overline{D}|c^t, s_{u,k}^{t^m})$ for all cells $c^t \in G$. This behavior is repeated until the next measuring time t_m exceeds the sensor end time $t_{u,k}^{end}$. The state of $s_{u,k}^t$ is received through SP input ports and consists of: the UAV state s_u^t for all types of sensors, and the sensor own state[13] for dynamics sensors.

4.5 Evaluator

This coupled model performs the remaining operations of the evaluation process: it updates the unobserved probabilities $p(c^t)$ and evaluates the utility-functions. As Fig. 7 show, it is broken down into Target coupled Model (TM) and the Evaluation Function (EF) atomic model, whose details are presented next.

4.6 Target Model (TM)

This coupled model estimates the state of the target that is being searched by the UAVs. Targets can remain static or move, and in the last case its simulation requires a probabilistic motion function to perform the prediction step in Eq. (1). Hence, as the TM structure changes to replicate both types of targets, Fig. 8 depicts the most complex structure of the TM model for dynamic targets.

4.6.1 Target Control (TC). The main function of this atomic model is to handle the operations of the unobserved probability map $p(c^t)$. Its behavior is depicted in Fig. 9 and Table 3. In more detail, after receiving the initial data and reaching the target start time t_τ^0, TM starts the target simulation, programming a δ_{int} by setting σ to the target end time t_τ^{end}. While the simulation lasts, $p(\overline{D}|c^t, s_{u,k}^{t_m})$ are received (due to the port coupling among the TC and sensor payload models), triggering the δ_{ext} transition function and updating $p(c^t)$ with Eq. (2). If the target is dynamic, the new $p(c^t)$ is sent to TMM (as it needs to be informed to predict the more recent $p(c^t)$). To do it, a δ_{int} is programmed by making $\sigma = 0$ and λ function output $p(c^t)$ before TC returns to its previous state. Predicted $p(c^t)$ can also be received from TMM, triggering a δ_{ext} that substitutes the previous $p(c^t)$ by the last one received. Besides, whenever $p(c^t)$ is updated due to the likelihoods, $P_d(t)$ and $ET(t)$ are also updated[14]. Afterwards, $p(c^t)$ is stored in the internal target-probability list. Finally, when t_τ^{end} is reached, an additional δ_{int} transition is programmed before TC outputs the result of the

[13]For instance the azimuth or elevation for a gimballed camera.
[14]They are not calculated at target predictions as $P_d(t)$ remains unchanged.

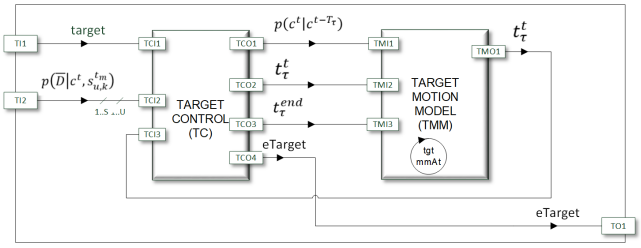


Figure 8: Target (TM) coupled model

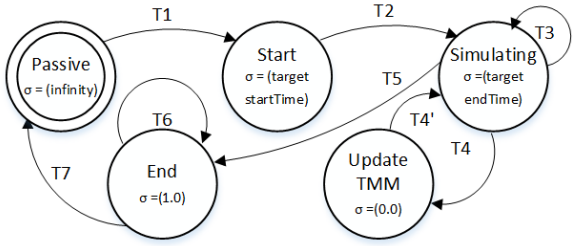


Figure 9: Target Control (TC) states diagram

Table 3: TC States Transitions

	δ_{int}	δ_{ext}
T1		At receiving initial data
T2	At reaching t_τ^0	
T3		When receiving a new $p(\overline{D}\|c^t, s_{u,k}^{t_m})$ or $p(c^t)$ from TMM
T4	To calculate new $p(c^t)$	
T4'		To send $p(c^t)$ to TMM
T6	At reaching t_τ^{end}	
T7		When new $p(\overline{D}\|c^t, s_{u,k}^{t_m})$ is received at t_τ^{end}
T8	end	

evaluation of the target model `eTarget`. This allows UM models to report their respective $p(\overline{D}|c^t, s_{u,k}^{t_m})$ when t_m is equal to t_τ^{end}.

4.6.2 Target Motion (TMM). This atomic model is only activated for dynamic targets and handles the operations of the target predictions. At the arrival of the initial data at t_τ^0, it calculates the motion model $p(c^t|c^{t-T_\tau})$ using the methods presented in [26]. Its λ function outputs periodically (at target motion rate) the $p(c^t)$ resulting from Eq. (1). It also receives $p(c^t)$ inputs from TC, triggering a δ_{ext} that updates the current $p(c^t)$. When TMM time is higher than t_τ^{end}, it transitions to a passive state.

4.7 Evaluator Function (EF)

This atomic model is executed at the end, after the UAV trajectories and target evaluations are available. It performs the remaining calculations to complete the evaluation process. That is, it computes the UAV collision and NFZ utility functions. Its behavior is simple: as soon as it receives $[\texttt{eUAV}]_{1:U}$ and `etarget`, it calculates the those utility functions and afterwards, it returns to a passive state.

Table 4: General characteristics of the two UAVs

Description	UAV1	UAV2
Flying height range (ft)	[500, 23000]	[500, 13100]
Flying speed range (kts)	[60, 110]	[115, 190]
Operating range (nm)	135	81
Flying autonomy (hours)	20	7
Payload	Radar	Camera(s)
Simulation flying height (ft)	3000	10000
Simulation flying speed (kts)	80	87.5

5 SIMULATIONS

In this section our M&S framework is used to evaluate different UAV target-search strategies for a real-world inspired scenario. The setup of all of them is a straightforward process, since after having defined the behavior and ports of all the models, their connection and configuration can be performed through configuration files. This characteristic is facilitated by the integral separation between the model and the simulation layers in the proposed MBSE&DES oriented methodology.

In the following, we firstly describe the mission, the system elements and their individual characteristics. Secondly, we propose and evaluate different UAV target-search alternatives to show the versatility of the tool. Finally, we analyze how the grid resolution affects the evaluation process.

5.1 Mission Description

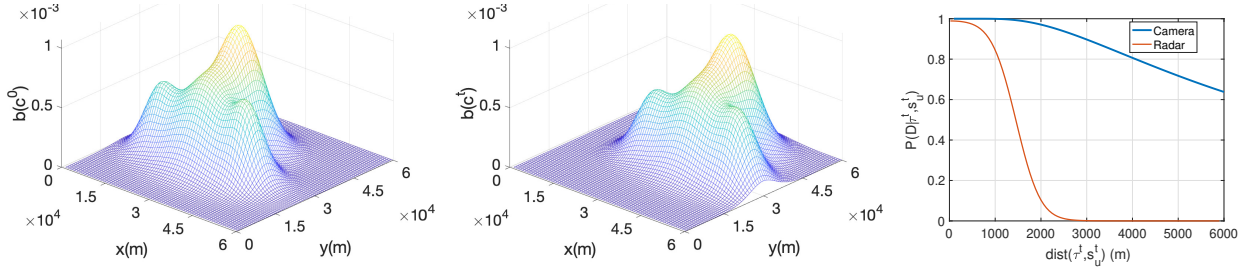
Our scenario is based on a search and rescue mission at the sea, where it is necessary to find a small drifting vessel (of 5 meters) with the survivors of a shipwreck close to the coast.

The search area Ω is a square of 30x30 nm^2, discretized in a grid of 80 x 80 cells to be able to define the initial target probability map $b(c^0)$, represented at Fig. 10(a), taking into account the last known position of the shipwreck, the sea wind and currents, and the time lapsed since the emergency call and the engagement of the first UAV in the search mission. As the vessel drifts due to the wind and sea currents, we also have to define $p(c|c^{t-T_\tau})$. The effect of our target motion model over $b(c^0)$, after applying Eq. (1) every T_τ = 375 seconds[15] during 2 hours, is displayed in Fig. 10(b).

Two fixed-wing UAVs, whose main characteristics are summarized in Table 4 are available for performing the search operation[16].

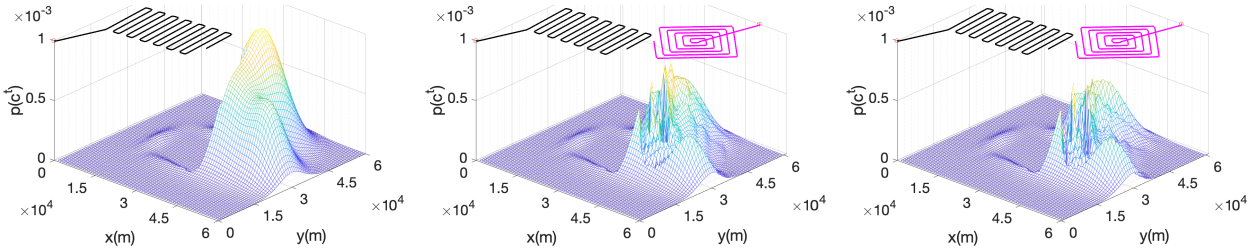
Besides, each UAV is equipped with a different type of sensor: a continuous wave radar and an electro-optical camera, this last mounted in a rotatory turret[17] that can re-orient the sensor during the mission. Besides, each sensor has a different likelihood $p(D|\tau^t, s_u^t)$ function and curve, which are presented (for the current scenario setup) as a function of the distance between τ^t and s_u^t in Fig. 10(c). Besides, the camera likelihood becomes zero outside of its footprint (550x550 m^2 at UAV2 flying altitude), whose location over G is modified by the UAV location and camera pose.

[15]This value is chosen considering the size of $c \in G$ and the wind/current velocities.
[16]In particular, the first UAV is inspired by the *Anonymized Real UAV Model*, while the second one is inspired by the one used in *Anonymized Real UAV Model*.
[17]Representative characteristics of the turret are a maximum slew rate of 60 degrees/s, an azimuth range of 360° and an elevation range from 90° to 0° (being 0° at the aircraft longitudinal axis).



(a) Initial belief $b(c^0)$ (b) Final $p(c^t)$ at t = 2 hours (c) Sensor likelihood

Figure 10: General information of the missions



(a) ST1- Single UAV (b) ST2 - Two UAVs, one camera (c) ST3 - Two UAVs, double camera

Figure 11: Simulation results of the versatility scenarios

Finally, it is worth noting that in the simulations of this paper, the UAV motion model is updated every 1 second and their control signal every 10 seconds, the radar and camera respectively provide information every 4 and 2 seconds, and the turret motion model is updated every 1 second. Besides, the UAV trajectories that are displayed in Fig. 11 and 13 have been obtained by creating a list of absolute heading setpoints to be applied at different moments of the mission[18].

5.2 Versatility Analysis

To demonstrate the versatility of our M&S framework, we analyze the performance of the following strategies:

- **Strategy 1 (ST1)** consists in making UAV1 (which arrives at Ω from the southwest corner at t_1^{start} = 0 seconds) follow a typical lawn-mower pattern during $t_{mission}^{end}$ = 7200 seconds.
- **Strategy 2 (ST2)** incorporates UAV2 (which arrives at Ω from the North at t_2^{start} = 1000 second), which has to perform a spiral pattern while the azimuth of its camera makes a (-90°,90°) zig-zag at fixed elevation (40°).
- **Strategy 3 (ST3)** incorporates a second camera to UAV2 whose azimuth makes a (-180°,180°) zig-zag at fixed elevation (40°).

The trajectory of each UAV[19] and the final unobserved probability $p(c^t)$ are displayed in the graphics of Fig. 11. Besides, the evolution of $P_d(t)$ is depicted in Fig. 12. In them, we can observe

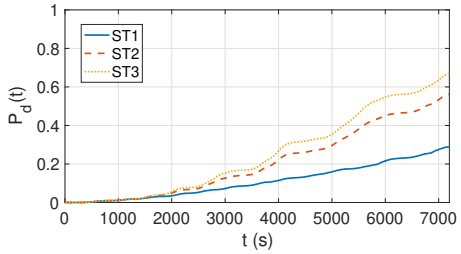


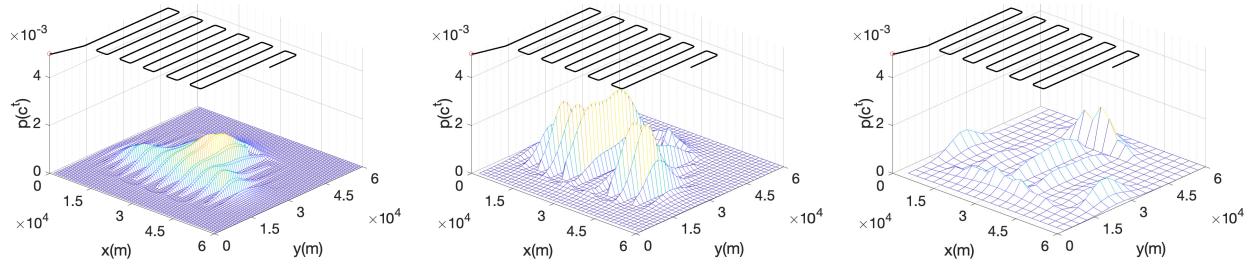
Figure 12: Detection probability in the versatility analysis

the expected behavior: as we add more sensors to the mission, less probability remains unobserved[20]. Besides, the improvement of $P_d(t)$ is not proportional to the number of sensors, as their likelihoods and ranges are different[21]. Finally, with the three strategies we have shown that the tool is capable of working with different number and types of UAVs and of sensors (including static and moving ones) and with models iterating asynchronously.

5.3 Resolution Analysis

The simulations in this section are used to take advantage of our evaluation tool to analyze the effect of the G resolution in the evaluation of the probability-based utility function. Three possible alternatives are tested:

[18] In other words, we are evaluating pre-defined UAV search patterns with our tool by defining an appropriated list of heading setpoints.
[19] The actual height of the trajectories is substituted by a fixed $p(c^t)$ value to simplify the graphics.
[20] Of course, when the new sensors are observing regions with probability to collect.
[21] In particular, although the camera footprint is smaller than the radar range, its movement make it collect more probability. Besides, the second camera can collect less, because it observes parts already observed by the first camera.



(a) RES1 - High Resolution (final ET= 7804 s) (b) RES2 - Intermediate Resolution (final ET= 6765 s) (c) RES3 - Low Resolution (final ET= 5541 s)

Figure 13: Results of the resolution analysis

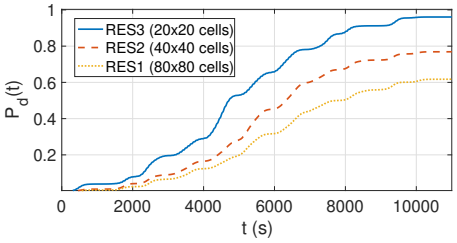


Figure 14: Detection probability in the resolution analysis

High resolution, with a grid of 80x80 cells, each of 750 x 750 m. Although it has the same initial belief $b(c^0)$ as the one in Fig.10(a), the target is static[22].

Middle resolution, with a grid of 40x40 cells, each of 1500 x 1500 m, whose values are obtained adding up the values of 2 x 2 consecutive cells of the high resolution belief $b(c^0)$.

Low resolution, with a grid of 20x20 cells, each of 3000 x 3000 m, whose values are obtained adding up the values of 2 x 2 consecutive cells of the middle resolution belief $b(c^0)$.

In this analysis we only use UAV1 during the whole mission and a list of heading setpoints that generates a lawn-mower pattern that leaves unobserved regions between two consecutive legs accordingly to the radar likelihood $P(D|\tau^t, s_u^t)$ represented in Fig. 10(c). Besides, as the assimilation step (stated at Eq. 2) requires $P(\overline{D}|c^t, s_u^t)$ and the radar likelihood $P(D|\tau^t, s_u^t)$ changes significantly within the cells of all the resolutions, we calculate $P(\overline{D}|c^t, s_u^t) = 1 - \sum_{\tau^t \in point(c^t, N_c)} P(D|\tau^t, s_u^t)/N_c$, spacing the evaluation points N_c in all the resolutions equally[23].

With this setup, we obtain the results represented in Fig. 13 and 14, which are clearly different at different resolutions[24]. This happens because at the lower resolution $P(\overline{D}|c^t, s_u^t)$ averages the values that will be put in several cells of the $p(c^t)$ of the higher resolution, consuming more unobserved probability that at the higher resolution. Hence, the probability of detection, in this example, is overestimated at the lower resolution[25]. Although increasing the precision can solve the problem, this is not always possible, as at a side effect the computational time is increased[26] , which can be critical when the evaluation framework is repeatedly used in an optimizer.

The previous analysis shows that our methodology not only allows us to easily set up complex scenarios, but it also facilitates the exhaustive analysis of the simulation results. Finally, the verification of models is also a straightforward with the different utilities provided by our DEVS simulation engine, as can be seen in [11].

6 CONCLUSIONS

This paper presents our approach to systematize the evaluation of UAV target-search strategies using MBSE and DES. To do it, we have develop a hierarchical DEVS framework that 1) models the evaluation process of a selected target-search planner and that 2) is adjustable to different number and types of UAVs and of sensors, and to static or moving targets. The iteration rates and events of the models are configurable too. The results show the versatility of the framework to analyze the effects of different UAV strategies and of the grid resolution in the evaluation of target-search missions.

As future work, we are thinking of expanding the functionality of the framework to let it evaluate other types of UAV trajectories (e.g. Dubin curves or splines) and/or use other types of probability models for the target (e.g. particles filters). We also plan to add an optimization module to the framework, following the same MBSE&DES principles, to be able to obtain automatically the best search strategies for the different UAVs. Finally, we will analyze the scalability of the proposed M&S framework, testing the simulation of scenarios with swarms of UAVs and larger exploration areas. To this end, we will perform both parallel and distributed simulations, with the help of the DEVS parallelization possibilities, as it has been already stated in [22].

REFERENCES

[1] S. Bernardini, M. Fox, and D. Long. 2017. Combining temporal planning with probabilistic reasoning for autonomous surveillance missions. *Autonomous Robots* 41 (2017), 181–203.

[22] Keeping the target static facilitates the comparison of the scenarios and lets us see that our framework can also work under a different configuration.

[23] That is, if to evaluate $P(\overline{D}|c^t, s_u^t)$ in a given resolution we use N_c points, we require $4xN_c$ points at the consecutive lower resolution.

[24] Note that the scale of the z-axis of Fig. 13 is not the same as the one in Fig. 10 and 11, because the beliefs of the other resolution accumulate more probability in each cell.

[25] The effect is the opposite in the expected time of detection (which is displayed at the captions of the graph of each solution): at lower resolutions it seems better (has a lower value) that at the highest.

[26] In particular, the computation time of RES1 is 69 s, of RES2 21 s and of RES3 10 s.

[2] F. Bourgault, T. Furukawa, and H. F. Durrant-Whyte. 2004. Decentralized Bayesian negotiation for cooperative search. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Vol. 3. 2681–2686.

[3] O. Breivik, A.A. Allen, C. Maisondieu, and M. Olagnon. 2013. Advances in search and rescue at sea. *Ocean Dynamics* 14 (2013), 9408–9428.

[4] F.M. Delle Fave, Z. Xu, A. Rogers, , and N. R. Jennings. 2010. Decentralised coordination of unmanned aerial vehicles for target search using the max-sum algorithm. In *Proceedings of the Workshop on Agents in Real Time and Environment*. 35–44.

[5] A. Fedorov. 2019. Path planning for UAV search using growing areaalgorithm and clustering. In *Fourth Conference on Software Engineeringand Information Managemen*.

[6] A.A. Feldbaum. 1961. Dual control theory III-IV. *Automn Remote Control* 21 (1961), 874–880.

[7] A.A. Feldbaum. 1961. Dual control theory I–II. *Automn Remote Control* 21 (1961), 874–880.

[8] J. Frost and L. Stone. 2001. *Review of search theory: advances and applications to search and rescue decision support*. Technical Report CG-D-15-01. US Coast Guard research and development center, Groton, CT, USA.

[9] B. Grocholsky, A. Makarenko, and H. Durrant-Whyte. 2003. Information-theoretic coordinated control of multiple sensor platforms. In *IEEE International Conference on Robotics and Automation (ICRA)*, Vol. 1. 1521–1526.

[10] S.B. Hall. 1997. A DEVS based simulation architecture for analysis of multi-vehicle interactions. In *Proceedings of SPIE - The International Society for Optical Engineering*. 287–294.

[11] K. Henares, J.L. Risco-Martín, J.L. Ayala, and R. Hermida. 2020. Unit testing platform to validate DEVS models. In *Proceedings of the Summer Simulation Conference*.

[12] J. Heo, J. Kim, and Y Kwon. 2018. Remote Operation SW for USV: Part I. Integrated Mission Planning System. *World Journal of Engineering and Technology* 6 (2018), 806–815.

[13] K. Holman, J. Kuzub, and G. Wainer. 2010. UAV search strategies using Cell-DEVS. In *Annual Simulation Symposium*. 192–199.

[14] J. Hu, L. Xie, J. Xu, and Z. Xu. 2014. Multi-agent cooperative target search. *Sensors* 14 (2014), 9408–9428.

[15] S. Ivić, B. Crnković, H Arbabi, S. Loire, P. Clary, and I. Mezić. 2020. Search strategy in a complex and dynamic environment: the MH370 case.

[16] L.P. Kaelbling, M.L. Littman, and A.R. Cassandra. 1998. Planning and acting in partially observable stochastic domains. *Artif. Intell.* 101 (1998), 99–134.

[17] D. Kingston, S. Rasmussen, and L. Humphrey. 2016. Automated UAV tasks for search and surveillance. In *IEEE Conference on Control Applications (CCA)*. 1–8.

[18] B Koopman. 1980. *Search and Screening: General Principles with Historical Applications*. Pergamon Press, Oxford, UK.

[19] T.M. Kratzke, L. Stone, and J. R. Frost. 2010. Search and Rescue Optimal Planning System. In *13th International Conference on Information Fusion*.

[20] P. Lanillos, S.K. Gan, E. Besada-Portas, G. Pajares, and S. Sukkarieh. 2014. Multi-UAV Target Search Using Decentralized Gradient-Based Negotiation with Expected Observation. *Information Science* 282 (2014), 92–110.

[21] L. Li, X. Zhang, W. Yue, and Z. Liu. 2021. Cooperative search for dynamic targets by multiple UAVs with communication data losses. *ISA Transactions* (2021).

[22] S. Mittal and J.L. Risco-Martín. 2017. DEVSML 3.0 stack: rapid deployment of DEVS farm in distributed cloud environment using microservices and containers. In *Proceedings of the 2017 Spring Simulation Multiconference*.

[23] A. Moreno, L. la Torre, J.L. Risco-Martín, E. Besada-Portas, and J. Aranda. 2011. DEVS-based validation of UAV plath planning in hostile environments. In *The international defense and homeland security simulation workshop*. 135–140.

[24] E. Pecker-Marcosig, S. Zudaire, M. Garrett, S. Uchitel, and R. Castro. 2020. Unified DEVS-based platform for modelling and simulation of hybrid control systems. In *Winter Simulation Conference*. 1051–162.

[25] S. Pérez-Carabaza, E. Besada-Portas, J.A. López-Orozco, and J.M. de la Cruz. 2016. A Real World Multi-UAV Evolutionary Planner for Minimum Time Target Detection. In *The Genetic and Evolutionary Computation Conference*. 981–988.

[26] S. Pérez-Carabaza, E. Besada-Portas, J.A. López-Orozco, and G. Pajares. 2019. Minimum Time Search in Real-World Scenarios Using MultipleUAVs with Onboard Orientable Cameras. *Journal of Sensors* 2019 (2019), 22.

[27] S. Pérez-Carabaza, J. Scherer, B. Rinner, J.A. López-Orozco, and E. Besada-Portas. 2019. UAV trajectory optimization for Minimum Time Search with communication constraints and collision avoidance. *Engineering Applications of Artificial Intelligence* 85 (2019), 357 – 371.

[28] M. Raap, M. Preuß, and S. Meyer-Nieberg. 2019. Moving target search optimization — A literature review. *Computers Operations Research* 105 (2019), 132 – 140.

[29] J. R. Riehl, G. E. Collins, and J. P. Hespanha. 2011. Cooperative Search by UAV Teams: A Model Predictive Approach using Dynamic Graphs. *IEEE Trans. Aerospace Electron. Systems* 47, 4 (2011), 2637–2656.

[30] K.M. Seo, C. Choi, T.G. Kim, and J.H. Kim. 2014. DEVS-based combat modeling for engagement-level simulation. *Simulation* 90, 7 (2014), 759–781.

[31] Simulink 2021. *Simulation and Model-Based Design. MathWorks*. Retrieved Feb 8, 2021 from https://www.mathworks.com/products/simulink.html

[32] R.D. Smallwood and E.J. Sondik. 1973. The Optimal Control of Partially Observable Markov Processes over a Finite Horizon. *Oper. Res.* 21 (1973), 1071–1088.

[33] L.D. Stone. 1975. *Theory of Optimal Search*. Academic Press, New York.

[34] J. Tisdale, Z. Kim, and J. K. Hedrick. 2009. Autonomous UAV path planning and estimation. *IEEE Robotics Automation Magazine* 16, 2 (2009), 35–42.

[35] G. Wainer. 2018. Advanced Cell-DEVS modeling applications: a legacy of Norbert Giambiasi. *Simulation* (2018), 1–27.

[36] G. Wainer and R. Castro. 2009. A Survey on the Application of the Cell-DEVS Formalism. *Journal of Cellular Automata* 10 (2009), 91–106.

[37] E.M. Wong, F. Bourgault, and T. Furukawa. 2005. Multi-vehicle Bayesian search for multiple lost targets. In *IEEE International Conference on Robotics and Automation*. 3169–3174.

[38] M.X. Zhang Y. Wang and Y.J. Zheng. 2017. A hyper-heuristic methodfor UAV search planning. In *International Conference on Swarm Intelligence*.

[39] Y. Yang, A.A. Minai, and M.M. Polycarpou. 2002. Decentralized cooperative search in UAV's using opportunistic learning. In *Proceedings of the AIAA Guidance, Navigation, and Control Conference and Exhibit*.

[40] P. Yao, Z. Xie, and P. Ren. 2019. Optimal UAV Route Planning for Coverage Search of Stationary Target in River. *IEEE Transactions on Control Systems Technology* 27, 2 (2019), 822–829.

[41] B.P. Zeigler, J.W. Marvin, and J.J. Cadigan. 2018. Systems Engineering and Simulation: Converging toward Noble Causes. In *Winter Simulation Conference*. 3742–3752.

[42] B.P. Zeigler, A. Muzy, and E. Kofman. 2018. *Theory of modeling and simulation: discrete event & iterative system computational foundations*. Academic Press.

[43] M. Zhang, J. Song, L. Huang, and C. Zhang. 2016. Distributed cooperative search with collision avoidance for a team of unmanned aerial vehicles using gradient optimization. *Journal of Aerospace Engineering* 30, 1 (2016).