An improved two-phase heuristic for active multistatic sonar network configuration

Owein Thuillier, Nicolas Le Josse, Alexandru-Liviu Olteanu, Marc Sevaux, Hervé Tanguy

An improved two-phase heuristic for active multistatic sonar network configuration

Owein THUILLIER[a,b,*], Nicolas LE JOSSE[a], Alexandru-Liviu OLTEANU[b], Marc SEVAUX[b], Hervé TANGUY[a]

[a] *Thales, Defence and Mission Systems, Brest, France*
{*owein.thuillier, nicolas.lejosse, herve.tanguy*}*@fr.thalesgroup.com*
[b] *Université Bretagne-Sud, Lab-STICC, UMR CNRS 6285, Lorient, France*
{*owein.thuillier, alexandru.olteanu, marc.sevaux*}*@univ-ubs.fr*

Abstract

An active sonar system consists of a source emitting a sound pulse (ping) and a receiver listening to the reflection of the wave on a target, known as the echo. Such a system is further divided into two distinct configurations. The first one, named monostatic, is made up of a collocated source and receiver, while the second one, referred to as bistatic, is based on a non-collocated source and receiver. To this extent, a Multistatic Sonar Network (MSN) is thus comprised of a set of sources and receivers deployed across a given Area of Interest (AoI), which, taken pairwise, form sonar systems in monostatic and/or bistatic configuration. In this paper, we therefore propose an efficient two-phase greedy heuristic to solve the Area Coverage (AC) problem in the scope of MSNs, a special case of Wireless Sensor Networks (WSNs), while taking into account existing coastlines. For this problem, the objective is to determine the optimal spatial layout of the MSN, i.e. the one that maximize the overall coverage of the AoI with regard to a limited number of sensors and a given probabilistic detection model. Furthermore, we use a Mixed Integer Linear Program (MILP) from the literature as a reference for the numerical experiments conducted on a dataset of diversified instances. The latter were specifically derived from Digital Elevation Models (DEMs) of AoIs selected throughout the globe and in such a way as to encompass a wide spectrum of peculiar geometric situations.

Keywords: OR in defense, multistatic sonar networks, area coverage problem, mixed integer linear program, constructive heuristic

*Corresponding author.
Email address: owein.thuillier@fr.thalesgroup.com (Owein THUILLIER)

1. Introduction

In the context of Anti-Submarine Warfare (ASW), sonar systems have been used extensively for decades as an effective means of searching, locating and tracking underwater threats. The expertise on these systems is therefore of importance for ASW decision-makers and the subject has naturally known a growth of interest in the literature throughout the years. In particular, the question of the optimal deployment of these sensors for coverage problems forms a flourishing research area, at the very forefront of the work conducted herein.

A sonar, acronym derived from "Sound NAvigation And Ranging", is a detection system that underwent considerable developments through the catalytic effect of the two successive World Wars. It relies substantially on the fact that acoustic waves propagate better underwater than any other type of physical wave (see (Urick, 1983; Cox, 1974; Lurton, 2002; Ainslie, 2010) for a more in-depth introduction to the subject). More specifically, there are two main types of sonar systems: passive and active. A passive sonar system is made up of a receiver listening to sounds radiated by a target, whereas an active sonar system consists of source emitting a sound pulse (ping) and a receiver listening to the reflection of the wave on the target, known as the echo (Urick, 1983). Note that the term transmitter is used interchangeably when referring to a source and that the abbreviations Tx and Rx are frequently used in the literature to designate a transmitter and a receiver respectively (Ngatchou et al., 2006; Washburn and Karatas, 2015).

Within the framework of this study, we focus solely on the case of active sonar systems and, to be more precise, we confine ourselves to the case of sonobuoys, a portmanteau of sonar and buoy. These are disposable acoustic units most often dropped in cylinder-shaped containers from an airborne carrier and which unfold upon impact with the water surface (Holler et al., 2008). That is, while the radio transmitter remains afloat, the core of the system is submerged at a predetermined depth that can range from a few dozen meters to several hundred meters depending on the use case (Holler et al., 2008; Iqbal et al., 2020). As mentionned in (Holler et al., 2008; Ozols and Fewell, 2011; Iqbal et al., 2020), a sonobuoy can be a source-only (e.g. ALFEA), a receiver-only (e.g. DIFAR) or a combined source-receiver (e.g. DICASS), commonly referred to as a post in the literature (Washburn, 2010; Ozols and Fewell, 2011; Washburn and Karatas, 2015; Craparo et al., 2019). Hence, as part of an ASW mission, several of these buoys may be deployed on a given Area of Interest (AoI) to form a surveillance network, thus acting as spotlights amidst the gloom in search of potential targets transiting in the vicinity or as a mere deterrent measure. The simplified operational context depicted here is illustrated in Figure 1 with a schematic representation of

the DICASS sonobuoy (Holler et al., 2008).

Active sonar systems are further divided into two distinct configurations depending on the source and receiver placement (Urick, 1983; Washburn, 2010; Hervé, 2011). The most prevalent configuration is the monostatic one in which a single sensor performs both the transmission and reception functions: the source and the receiver are said to be collocated (i.e. a post). In contrast, when the emission and the reception are carried out by two distinct sensors, the configuration is then referred to as bistatic: the source and the receiver are said to be non-collocated, or delocalized. These two configurations are geometrically portrayed in Figure 2.

To this extent, a Multistatic Sonar Network (MSN) is thus comprised of a set of sources and receivers deployed across a given AoI, which, taken pairwise, form sonar systems in monostatic and/or bistatic configuration. A network of this kind presents numerous advantages over the inherently more restrictive monostatic sonar networks (Cox, 1989). For example, it becomes a much more arduous task for a potential threat to counter-detect a receiver since it remains silent and it may be deployed as a standalone unit. Also, given that receivers are cheaper than sources (Amanipour and Olfat, 2011; Washburn and Karatas, 2015), it is therefore possible to cover larger areas at a reduced cost compared to a monostatic sonar network where, by definition, as many sources as receivers must be deployed. That being said, the chief disadvantage of MSNs lies in the complex and unusual geometry of the coverage area (Cox, 1989; Karatas, 2013) induced by the use of sonar systems in bistatic configuration. This makes the performance evaluation of such a network mathematically more challenging. In another vein, it is also important to mention that MSNs share a certain proximity with Multistatic Radar Networks (MRNs), as they relies on similar mathematics (Washburn, 2010).

In the literature related to the deployment in Wireless Sensor Networks (WSNs), MSNs being a special case of WSNs, we find the following three main types of coverage problems: "Barrier Coverage (BC)", "Point (target) Coverage (PC)" and "Area (blanket) Coverage (AC)" (see (Wang, 2011; Khoufi et al., 2017; Elhabyan et al., 2019) for an extended overview of these problems). Note that in some papers related to WSNs, the AoI is sometimes indiscriminately termed Region of Interest (RoI) (Mohamed et al., 2018) or Field of Interest (FoI) (Tripathi et al., 2018). In this paper, we therefore propose an efficient two-phase greedy heuristic to solve the AC problem in the scope of MSNs while taking into account existing coastlines. For this problem, the objective is to determine the optimal spatial layout of the MSN, i.e. the one that maximizes the overall coverage (in the sense of insonification) of the AoI with regard to a limited number of sensors and a

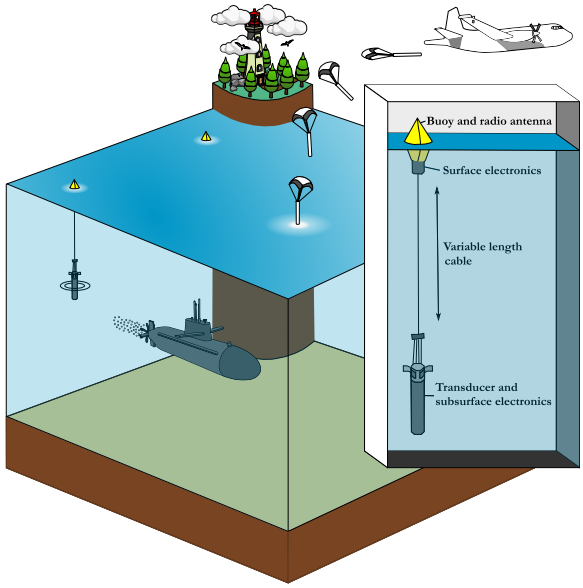

Figure 1: A simplified illustration of the operational context.

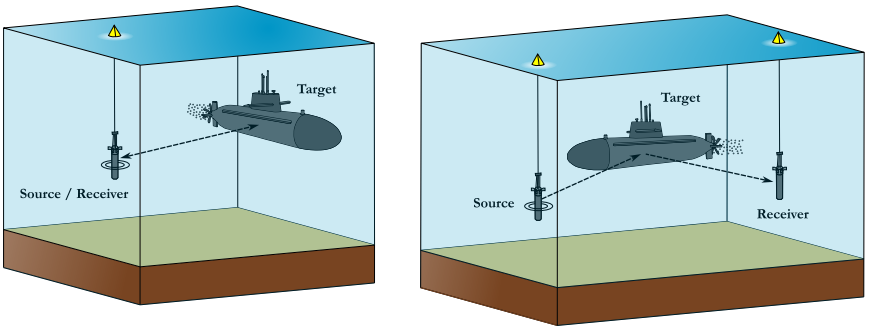

Figure 2: Monostatic (left) and bistatic (right) configurations.

given probabilistic detection model. This could be either for the purpose of searching a target over a large area, or to protect the vicinity of some High Value Units (HVUs), for example. The overall coverage corresponds here to what can be found under the generic term Measure of Effectiveness (MoE) in some of the MSN-specific literature (Been et al., 2007; Strode et al., 2012).

More precisely, as done in (Craparo and Karatas, 2020; Fügenschuh et al., 2020), we approach the AC problem by reducing it to a PC-type problem in order to have a discrete approximation of the surface to be covered. That is to say, as illustrated in Figure 3, an AoI is first identified (a), isolated (b) and then discretized (c) by means of a Digital Elevation Model (DEM) (Guth et al., 2021), i.e. a regular rectangular grid where the maritime cells are the ones with an elevation less than or equal to zero (arbitrary choice). Then, we place a Point of Interest (PoI) (Khoufi et al., 2017), called here a "target" to be consistent with the literature (Craparo et al., 2017), and a deployment position in the center of each maritime cell of the resulting grid (d). Note that a target refers to a physical position that we wish to monitor (whose location is known beforehand) and not to an object in the strict sense. Indeed, having no information on the real target(s), these are dummy/control targets allowing us to evaluate the performance of the network on a discrete set of positions (neglecting the orientation of the target). In this manner, a cell is said to be covered if and only if the target in its center is considered as detected (or covered) by the current network (see Section 3 for more details), which is something that is regularly done in the literature related to WSNs (Khoufi et al., 2017).

Moreover, if greater precision is required, the resolution of the current grid may be artificially increased by performing an upsampling procedure, i.e. by subdividing each cell until the desired resolution is reached and then performing an interpolation (e.g. nearest neighbor, bilinear or cubic). Or, in a similar way, it is possible to reduce the resolution of the current grid by performing a downsampling procedure, i.e. by dragging a filter of fixed size over the grid and aggregating neighboring cells using an operator (e.g. max, min or mean). This procedure could make it possible to tackle larger areas, taking care however to remain consistent with the sonar detection range (i.e. to have at least one target in range).

Now, suppose that we consider only posts (source and receiver collocated) and that we do not take into account the interactions between posts, then this restricted problem can be seen as a Maximum Coverage Location Problem (MCLP) (Church and Revelle, 1974; Megiddo et al., 1983) where the facilities are the sensors and the clients the different targets. The latter problem has been proved $\mathcal{NP}$-Hard (Megiddo et al., 1983) and our problem is at least as complicated because of the interactions between the facilities

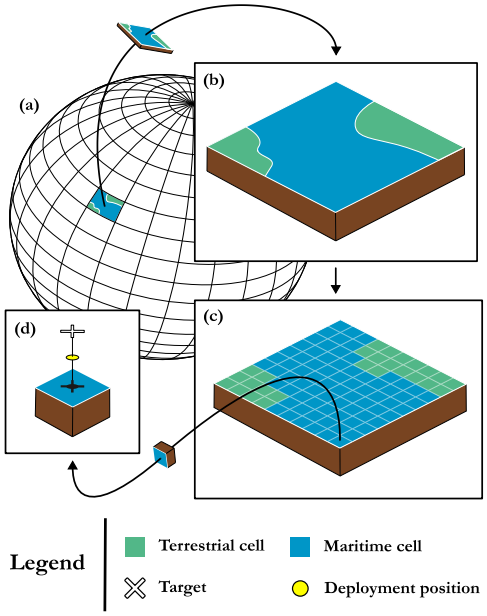

Figure 3: Transformation of an AC-type problem to a PC-type problem.

(the sensors). As a result, exact techniques are not suitable to handle real life instances, thus motivating the derivation of a heuristic for the problem at hand.

Regarding the assumptions considered in the scope of this study, we first consider that the sensors and targets are stationary (refer to (Grasso et al., 2013) for work on the deployment of sonobuoys with oceanic drift in a monostatic case) and evolve in a two-dimensional space with homogeneous environmental conditions. Additionally, we require that at most one source and at most one receiver may be deployed on a given deployment position (which would then correspond to a post). Lastly, the sensors are assumed to be homogeneous (identical performances) and omnidirectional, which is relevant to our case study: the sonobuoys.

The contributions of this research work can be summarized below:

- Creation and distribution of a benchmark of 27 instances derived from pre-processed open access elevation data (bathymetric and altimetric measurements).
- Proposal of the first heuristic, to the best of our knowledge, that solves Area Coverage (AC), Barrier Coverage (BC) and Point Coverage (PC) problems for MSNs while taking into account coastlines.

- Taking into account binary and probabilistic models as well as the direct blast effect often neglected in the literature.
- Comparison of this heuristic with the best model in the literature which has been re-implemented in this paper.
- Construction of an original visualization tool for the solution output.

The paper is organized as follows. Section 2 contains a literature review on coverage problems in multistatic networks. Section 3 consists of a formal description of the problem considered, including some notions of multistatic detection theory and how coastlines are handled. Section 4 is dedicated to the introduction of a Mixed Integer Linear Program (MILP) from the literature while Section 5 concerns the two-phase greedy heuristic developed as part of this work. Section 6 contains numerical experiments on the heuristic using the MILP model as a reference and a state-of-the-art solver. Finally, Section 7 contains a conclusion of the work that has been carried out so far as well as some perspectives for further research.

2. Literature Review

Area Coverage (AC). Ngatchou et al. (Ngatchou et al., 2006) use a multiobjective Particle Swarm Optimization (PSO) algorithm to determine sensor placement by maximizing coverage and minimizing the number of sensors required simultaneously. DelBalzo and Stangl (DelBalzo and Stangl, 2009) use a genetic algorithm to optimize both the placement of individual sensors (depth included) and the emission times of individual sonobuoys in a non-homogeneous environment (known as SCOUT). Ozol and Fewell (Ozols and Fewell, 2011) studied a total of 27 different geometric patterns in open water (i.e. no coastlines) to find the most efficient one for large area coverage. Strode et al. (Strode et al., 2012) use a genetic algorithm to determine the positions of various sensors. Washburn and Karatas (Washburn and Karatas, 2015) derive an analytical theory to predict the probability of detection based on randomly deployed sensors and use it to determine optimal patterns. Building on this work, Karatas and Craparo (Karatas and Craparo, 2015) use simulations to quantify the impact of the direct blast effect on coverage. Karatas et al. (Karatas et al., 2016) use simulations to quantify the coverage of a mobile source performing parallel scans in a stationary receiver field. Fugenschuh et al. (Fügenschuh et al., 2020) propose several models with different linearizations and compare them in the context of two problems: maximizing the total area covered with a limited number of sensors (problem studied here) and minimizing the economic costs associated with the deployment of the various sensors in order to cover the entire area.

Barrier Coverage (BC). In MRNs, Gong et al. (Gong et al., 2013) propose a method for optimal radar placement on a line segment in order to maximize the worst-case intrusion detectability. Li et al. (Li et al., 2021) study the Circular Barrier Coverage (CBC) problem and propose a method based on the "equipartition strategy", with the objective of determining the optimal deployment patterns of multistatic radar for a sub-problem. The latter patterns are then used through an Integer Linear Program (ILP) and an exhaustive method to address the global problem. Li et al. (Li et al., 2022) propose an ILP and exhaustive search to determine multiple unequal-width barrier coverages of the AoI (with different deployment sequences).

Point Coverage (PC). Craparo et al. (Craparo et al., 2017) propose an algorithm named Divide Best Sector (DiBS) for the placement of a single source when a number of receivers are already deployed (with an iterative extension for the placement of multiple sources). This last paper is essentially based on the works present in (Kuhn, 2014). (Craparo et al., 2019) propose for the first time the optimal placement of sources and receivers for this type of problem in open water (without coastlines). They propose two ILPs (DISC-LOC-M and DISC-LOC-ENUM) as well as two heuristics: Adapt-LOC and Iter-LOC, based on a procedure named LOC-GEN-II, which is an enhanced version of LOC-GEN (Craparo and Karatas, 2018) that may be found initially in (Hof, 2015). Craparo and Karatas (Craparo and Karatas, 2020) propose an exact OPT-LOC solution method and a greedy GREEDY-LOC heuristic for source placement when receivers are already deployed. This is an extended form of the work of Craparo and Karatas (Craparo and Karatas, 2018).

3. Problem Formulation

3.1. Formal definition

Let $m \in \mathbb{N}^+$ be the number of maritime cells in the grid. We then have $T = \{t_1, \ldots, t_m\} \subseteq \mathbb{R}^2$ the set of targets positions and $E = \{e_1, \ldots, e_m\} \subseteq \mathbb{R}^2$ the set of deployment positions, i.e. a target and a deployment position in the center of each maritime cell. Besides, we also have $n_s \in \mathbb{N}^+$ and $n_r \in \mathbb{N}^+$ corresponding respectively to the number of sources and receivers available. In order to compare with the literature, we directly limit the number of sources and receivers, but the extension to a limited number of buoys that are source-receiver, source-only and receiver-only is rather straightforward.

An admissible solution for the problem under consideration is then a network $\omega = (S, R)$ with $S \subseteq E$ the set of sources positions and $R \subseteq E$ the set of receivers positions such that $|S| \leq n_s$ and $|R| \leq n_r$. The set of all

possible networks (admissible solutions) is therefore

$$\Omega = \{(S, R) \mid S, R \subseteq E \wedge |S| \leq n_s \wedge |R| \leq n_r\}. \tag{1}$$

As a reminder, a sonar system is defined as a source-receiver pair in monostatic or bistatic configuration. Thus, the set of all possible sonar systems is

$$\Xi = \left\{(s, r) \in E^2\right\}. \tag{2}$$

By extension, for a given network $\omega \in \Omega$, we write $\Xi_\omega \subseteq \Xi$ the set of all sonar systems of ω. Naturally, we have $|\Xi_\omega| \leq n_s n_r$.

In addition, we note $P_d^{(s,r)}(t)$ the instantaneous detection probability of a target $t \in T$ by a sonar system $(s, r) \in \Xi$ and $P_d^\omega(t)$ the cumulative detection probability of a target $t \in T$ by a network $\omega \in \Omega$, here calculated as the probability that at least one of the sonar systems $(s, r) \in \Xi_\omega$ detects the aforementioned target (Subsection 3.2 will describe the computation of the detection probabilities). Furthermore, as done in (Fügenschuh et al., 2020), we consider that a target $t \in T$ is detected by a network $\omega \in \Omega$ when the cumulative detection probability $P_d^\omega(t)$ is greater than or equal to a threshold $\phi \in [0, 1]$ set upstream and generally close to 1 (e.g. $\phi = 0.95$).

The objective function f used to evaluate a network $\omega \in \Omega$ here corresponds to the proportion of covered cells (i.e. detected targets) or, in other words, the coverage rate. It is defined as

$$\begin{aligned} f &: \Omega \to [0, 1] \\ &\omega \mapsto \frac{1}{|T|} \sum_{t \in T} I\left(P_d^\omega(t)\right), \end{aligned} \tag{3}$$

where

$$I(x) = \begin{cases} 1 & \text{if } x \geq \phi, \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

Note that it is possible to weight the targets in such a way as to accentuate the importance of some cells over others by introducing a reward for the detection of the target $t \in T$ (and by extension for the coverage of the cell in which it is located). This may make sense when protecting HVUs, for example. On the other hand, this weighting can also be seen as a way to model a probability distribution of the target presence in the AoI, which might make sense when searching for a target (supposing prior knowledge). However, in the remainder of this study and for the sake of simplicity, we assume a unitary reward for all targets.

Finally, we look for the optimal network $\omega^* \in \Omega$, i.e. the one maximizing

the objective function f:

$$\omega^* = \arg\max_{\omega \in \Omega} f(\omega)\,. \tag{5}$$

3.2. Multistatic detection theory

In the case of a sonar in bistatic configuration, let $d_{t,s} \in \mathbb{R}^+$ the distance separating the target $t \in T$ from the source $s \in E$ and $d_{t,r} \in \mathbb{R}^+$ the distance separating the target $t \in T$ from the receiver $r \in E$. Then, by solving the active sonar equations for the distance-dependent transmission loss term while neglecting frequency-related absorption losses gives us the following equality (Urick, 1983; Cox, 1989):

$$d_{t,s} d_{t,r} = \rho_0^2\,, \tag{6}$$

where ρ_0 (in km) is referred to as Range of the Day (RoD) (Fewell and Ozols, 2011), range of the moment (Abbot and Dyer, 2002) or r_{50} (Ainslie, 2010). It is defined by convention (Jensen et al., 2011) as the distance at which the probability of detecting a target is 50% for a sonar system in a monostatic configuration and in a given environment.

More precisely, this last equality defines isocontours of constant detection probability describing what is commonly called "Cassini ovals" (Cox, 1989; Karatas, 2013) and whose interior region is defined by the set of points for which the probability of detection is greater than or equal to 50%. As illustrated in Figure 4, the different shapes of a Cassini oval can be classified into several categories according to the ratio $\frac{d_{s,r}}{\rho_0}$, where $d_{s,r}$ is defined as the inter-sensor spacing. First, for a sonar system in monostatic configuration, i.e. when $\frac{d_{s,r}}{\rho_0} = 0$, we obtain a circle centered on the sonar system (i.e. a post) and of radius ρ_0 (a). Secondly, for a sonar system in bistatic configuration, several cases may be considered. For $\frac{d_{s,r}}{\rho_0} \leq \sqrt{2}$, we get an ellipse (b), while when $\sqrt{2} < \frac{d_{s,r}}{\rho_0} < 2$, we get a a kind of bone (peanut) shape (c). Furthermore, a particular shape called Bernoulli lemniscate is reached when $\frac{d_{s,r}}{\rho_0} = 2$ (d), before finally ending on two disjoint ovoids when the sensors are far enough from each other, i.e. whenever $\frac{d_{s,r}}{\rho_0} > 2$ (e,f).

Then, as argued by Fewell and Ozols (Fewell and Ozols, 2011), the link with the monostatic case is made by introducing the distance in monostatic equivalent $\rho_{t,s,r} = \sqrt{d_{t,s} d_{t,r}}$. This way, it is then possible to refer to the detection curves obtained for a sonar in monostatic configuration and to generalize it by allowing the source and the receiver to be separated from one another. Within the scope of this study, we will use schematic detection curve chosen to model many different situations and such that we have $P_d^{(s,r)}(t) = 0.5$ when $\rho_{t,s,r} = \rho_0$ to be in line with the RoD definition. More

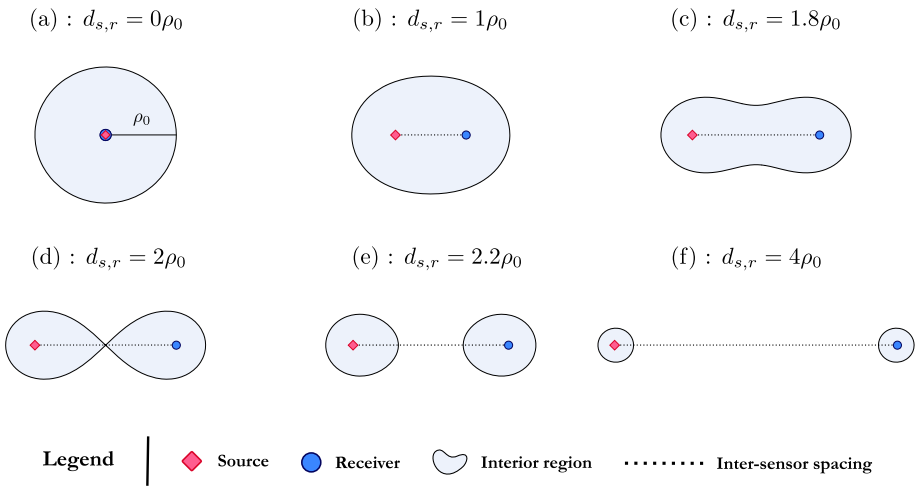

Figure 4: Typical Cassini ovals.

realistic detection curves are generally non-monotonous (Fewell et al., 2008; Fewell and Thredgold, 2009; Fewell and Ozols, 2011), due amongst other things to the refraction of acoustic rays in the different strata of the water column and to the seabed topology.

That being said, and although there are other schematic detection curves in the literature, in this paper we will restrict ourselves to the class of functions used by Fewell and Ozols (Fewell et al., 2008; Fewell and Ozols, 2011) and named Fermi because of its proximity to the Fermi–Dirac distribution (McDougall and Stoner, 1938). It is a logistic function (sigmoid) used to compute the instantaneous detection probability of a target $t \in T$ by a sonar system $(s, r) \in \Xi$ and it is defined as

$$P_d^{(s,r)}(t) = \frac{1}{1 + 10^{\left(\frac{\frac{\rho_{t,s,r}}{\rho_0} - 1}{b}\right)}}. \tag{7}$$

For this class of function, the parameter $b \in \mathbb{R}^+$ called diffusivity parameter enables to control the rate at which the detection probability fades as $\rho_{t,s,r}$ increases. Moreover, when $b \to 0$, this probabilistic (diffuse) model approaches a deterministic (binary) model called "definite-range law" or "cookie-cutter" detector in the MSN literature (Fewell and Ozols, 2011). These detection models are illustrated in Figure 5 where the detection probability $P_d^{(s,r)}(t)$ is expressed as a function of the distance in monostatic equivalent $\rho_{t,s,r}$ multiple of the RoD ρ_0.

Finally, let us consider a network $\omega \in \Omega$ and assume that each detection is stochastically independent from one another. Hence, the cumulative

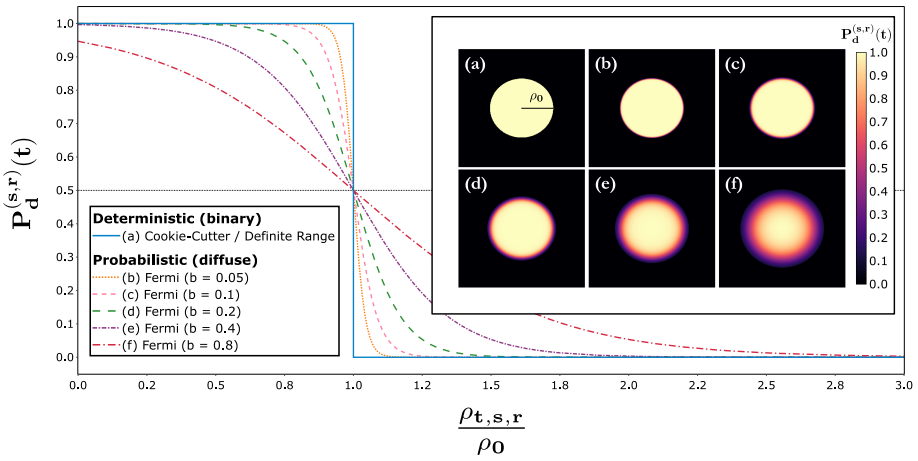

Figure 5: Schematic reference curves for a sonar system in monostatic configuration.

probability of detection, here expressed as the probability that at least one of the sonar systems $(s, r) \in \Xi_\omega$ detects a target $t \in T$, is defined as

$$P_d^\omega(t) = 1 - \prod_{(s,r)\in\Xi_\omega} \left(1 - P_d^{(s,r)}(t)\right) , \tag{8}$$

which is equal to one minus the probability that none of the sonar systems $(s, r) \in \Xi_\omega$ detects the target $t \in T$.

Nevertheless, in practice, there is a masking zone between the source and the receiver within which there is no detection (Cox, 1989; Fewell and Ozols, 2011; Karatas and Craparo, 2015), also called "dead zone" and ellipsoidal in shape. This is known as the direct blast effect and it occurs when the signal reflected by a target (the echo) arrives at the receiver at the same time as a portion of the ping coming directly from the source and is therefore partially masked. The size of the blind zone is directly related to the transmission time $\tau \in \mathbb{R}^+$ (in seconds) of the ping traveling at celerity $c \in \mathbb{R}^+$ (in $km \cdot s^{-1}$). Let $r_b \in \mathbb{R}^+$ equal to half the transmitted "pulse length" or "pulse width", i.e. $r_b = \frac{c\tau}{2}$ (in km) (Fewell and Ozols, 2011). Hence, without going into the mathematical details, there will be no detection ($P_d^{(s,r)}(t) = 0$) if

$$d_{t,s} + d_{t,r} < d_{s,r} + 2r_b \, . \tag{9}$$

Many papers in the literature do not take this effect into account (Craparo et al., 2017; Craparo and Karatas, 2018; Craparo et al., 2019; Craparo and Karatas, 2020), as it can be greatly reduced through signal processing (or "pulse compression"), as shown in (Cox, 1989; Fewell and Ozols, 2011). However, the heuristic introduced in this paper will be compatible with the

consideration of the direct blast effect.

3.3. Coastline management

For coastline management, the idea is based on the principle exposed by Amanatides et al. in (Amanatides and Woo, 1987). Indeed, these authors propose an efficient algorithm to discretize a segment through a grid in the context of ray-tracing, heavily used in image synthesis. Thus, in our case, for a given target $t \in T$ and a sonar system $(s, r) \in \Xi$, it is sufficient to draw two segments, one between the source and the target and the other between the receiver and the target. If one of the two segments intersects a cell with a positive elevation (terrestrial cell), then the probability of detection is set to zero ($P_d^{(s,r)}(t) = 0$).

4. Mathematical Programming Formulation (MILP)

In the paper by Fügenschuh et al. (Fügenschuh et al., 2020), two mathematical formulations are presented as being the most efficient without one standing out from the other for the AC problem herein under consideration. More specifically, these are the two formulations based on the linearizations of Oral and Kettani (Oral and Kettani, 1992) and Chaovalitwongse et al. (Chaovalitwongse et al., 2004). Although there are other formulations and sometimes several variants, only the formulation relying on the linearization of Oral and Kettani in its most efficient variant noted OK1-S will be introduced in the following.

First, for a network $\omega \in \Omega$, let us recall that a target $t \in T$ is considered detected if the cumulative probability of detection $P_d^{\omega}(t)$ is greater than or equal to the threshold ϕ:

$$P_d^{\omega}(t) = 1 - \prod_{(s,r)\in\Xi_{\omega}} \left(1 - P_d^{(s,r)}(t)\right) \geq \phi\,. \tag{10}$$

Following an approach proposed by Prim (Prim, 1957), this latter constraint may be linearized by taking its logarithm as follows:

$$\sum_{(s,r)\in\Xi_{\omega}} \log_{(1-\phi)} \left(1 - P_d^{(s,r)}(t)\right) \geq 1\,, \tag{11}$$

where $\log_{(1-\phi)} \left(1 - P_d^{(s,r)}(t)\right)$ corresponds to the individual contribution of the sonar system $(s, r) \in \Xi_{\omega}$ in the detection of the target $t \in T$.

Then, we set $\forall e \in E, \forall t \in T$

$$L_{e,t} = \sum_{e'\in E} \log_{(1-\phi)} \left(1 - P_d^{(e,e')}(t)\right)\,, \tag{12}$$

where $L_{e,t}$ corresponds to an aggregation of the individual contributions $\log_{(1-\phi)}\left(1 - P_d^{(e,e')}(t)\right)$ for a fixed deployment position $e \in E$ (which may accommodate a source) and target $t \in T$. This is the OK1-S variant, as the aggregation is done by fixing a deployment position that may accommodate a source. Note that, alternatively, if the aggregation of individual contributions is done by fixing a position $e' \in E$ that may accommodate a receiver, we thus obtain the variant called OK1-R by the authors (Fügenschuh et al., 2020).

Furthermore, we introduce the following decision variables: $x_t \in \{0,1\}$ with $x_t = 1$ if the target $t \in T$ is detected, $s_e \in \{0,1\}$ with $s_e = 1$ if a source is deployed on the position $e \in E$ and, finally, $r_e \in \{0,1\}$ with $r_e = 1$ if a receiver is deployed on the position $e \in E$.

We also have $\forall e \in E, \forall t \in T$, the auxiliary variable $z_{e,t} \in \mathbb{R}^+$ which can be defined as

$$z_{e,t} = \left(L_{e,t} - \sum_{e' \in E} \log_{(1-\phi)}\left(1 - P_d^{(e,e')}(t)\right) r_{e'} \right) s_e \,. \tag{13}$$

Since $\sum_{e \in E} \sum_{e' \in E} \log_{(1-\phi)}\left(1 - P_d^{(e,e')}(t)\right) s_e r_{e'} = \sum_{e \in E} L_{e,t} s_e - z_{e,t}$, it is then sufficient to linearize this auxiliary variable to only take into account the individual contributions when $s_e r_{e'} = 1, \forall (e,e') \in E^2$.

Finally, the Mixed Integer Linear Programming (MILP) named OK1-S is written as follows:

$$\text{maximize} \quad \frac{1}{|T|} \sum_{t \in T} x_t \tag{14}$$

$$\text{subject to} \quad \sum_{e \in E} (L_{e,t} s_e - z_{e,t}) \geq x_t \qquad \forall t \in T \tag{15}$$

$$z_{e,t} \geq L_{e,t} s_e - \sum_{e' \in E} \log_{(1-\phi)}\left(1 - P_d^{(e,e')}(t)\right) r_{e'} \quad \forall e \in E, \forall t \in T \tag{16}$$

$$\sum_{e \in E} s_e \leq n_s \tag{17}$$

$$\sum_{e \in E} r_e \leq n_r \tag{18}$$

$$x_t \in \{0,1\} \qquad \forall t \in T \tag{19}$$

$$s_e \in \{0,1\} \qquad \forall e \in E \tag{20}$$

$$r_e \in \{0,1\} \qquad \forall e \in E \tag{21}$$

$$z_{e,t} \in \mathbb{R}^+ \qquad \forall e \in E, \forall t \in T \tag{22}$$

To understand this modeling, we must distinguish two cases.

- In the first case, for a fixed deployment position $e \in E$ and target $t \in T$, we assume that $s_e = 0$. If $s_e = 0$, then the auxiliary variable $z_{e,t}$ is bounded inferiorly by $-\sum_{e' \in E} \log_{(1-\phi)} \left(1 - P_d^{(e,e')}(t)\right) r_{e'}$ in (16). This last term being strictly negative, we thus have $z_{e,t} \geq 0$ by (22). Moreover, knowing that we wish to maximize the number of detected targets in (14), we therefore try to maximize $\sum_{e \in E}(-z_{e,t})$ in (15), which is the same as minimizing $z_{e,t}$ and thus to obtain $z_{e,t} = 0$. That is, if $s_e = 0$ then no individual contributions is taken into account, which is correct because we have $s_e r_{e'} = 0, \forall e' \in E$.

- In the second case, suppose that $s_e = 1$ for a fixed deployment position $e \in E$ and target $t \in T$. If $s_e = 1$, then the auxiliary variable $z_{e,t}$ is bounded inferiorly by $L_{e,t} - \sum_{e' \in E} \log_{(1-\phi)} \left(1 - P_d^{(e,e')}(t)\right) r_{e'}$ in (16). Now, this last term being strictly positive, we have $z_{e,t} \geq L_{e,t} - \sum_{e' \in E} \log_{(1-\phi)} \left(1 - P_d^{(e,e')}(t)\right) r_{e'}$ and, as before, by (14) and (15), we then have $z_{e,t} = L_{e,t} - \sum_{e' \in E} \log_{(1-\phi)} \left(1 - P_d^{(e,e')}(t)\right) r_{e'}$ since we try to minimize $z_{e,t}$. This gives us $L_{e,t} - z_{e,t} = \sum_{e' \in E} \log_{(1-\phi)} \left(1 - P_d^{(e,e')}(t)\right) r_{e'}$ in (15), which is equivalent to accounting for individual contributions when $s_e r_{e'} = 1, \forall e' \in E$.

The deployment constraints (17) and (18) compel the solution to use only as many sources (n_s) and receivers (n_r) as available. The constraints (19), (20) and (21) are integrity constraints on the decision variables and (22) a positivity constraint on the auxiliary variable.

5. Heuristic

In this section, we will present the two-phase greedy heuristic designed to construct iteratively an admissible solution (a network) of good quality and in a reasonable time for the AC problem. We begin first with the naïve version of the heuristic and then the improved version which consists of a succession of various improvements. The decision-making criteria used for these heuristics is identical to that used in the MILP formulation: maximizing the area covered with a limited, known beforehand, number of sensors (hard constraint). Furthermore, given that there are no economic costs for sources or receivers, it makes perfect sense to deploy them all as long as the area is not fully covered. At each iteration, we are only interested in the incremental gain in coverage rate resulting from the deployment of a sensor.

5.1. Naïve Approach

In the first phase, we place as many posts as possible, i.e. $\min(n_s, n_r)$ collocated source–receiver pair. In particular, at each iteration, we place one source and one receiver on the deployment position that locally maximize the coverage rate. At the end of this phase, if we still have sensors to be placed, then we have only sources or receivers.

During the second phase, we place as many sensors as possible of the remaining type, i.e. $\max(n_s, n_r) - \min(n_s, n_r)$. Following the same principle as before, at each iteration, we place either a source or a receiver on the deployment position that locally maximize the coverage rate (see algorithm 1 for details).

The advantage of proceeding in this way is that it is then sufficient to probe only one position at a time, which gives us a time complexity of $\mathcal{O}(|E||T||S||R|)$ against a complexity of $\mathcal{O}(|E|^2|T||S||R|)$ if we intended to place non-collocated source-receiver pair at each iteration. Hence, with $m = |T| = |E|$ (one deployment position and one target per maritime cell) and assuming that $|S| + |R| \ll m$, we obtain a greedy heuristic with time complexity of $\mathcal{O}(m^2)$.

The heuristic stops when all the sensors have been deployed, this is the primary stopping criterion. Other stopping criteria are also possible such as a maximum computational budget $T_{max} \in \mathbb{R}^+$ (in seconds) and a minimum marginal gain $\sigma \in [0, 1]$, i.e. the proportion of new covered cells (detected targets) at each iteration.

5.2. Improved Approach

The improved version of the heuristic consists of a succession of improvements explained hereafter. We draw attention to the fact that the improvements will be given here in a more general case, i.e. where the deployment positions do not necessarily coincide with the target positions. However, when possible, specific improvements will be given for the particular case that is addressed here. The complete pseudo-code is available in Appendix A.

5.2.1. Memoïzation

Knowing that we iteratively add either a post or a single sensor to the network $\omega \in \Omega$, it is then redundant to recompute all the detection probabilities for each of the sonar systems $(s, r) \in \Xi_\omega$. Indeed, if we introduce a matrix $\overline{P}_d^\omega \in M_{1,|T|}([0, 1])$ storing the non-detection probabilities for each of the targets $t \in T$ by the current network $\omega \in \Omega$, it is then sufficient to compute only the contributions related to the newly formed sonar systems.

Algorithm 1: Two-phase greedy heuristic: naïve version

Initialization
$S \leftarrow \emptyset$; $R \leftarrow \emptyset$; $\omega \leftarrow (S, R)$
Phase 1 ; /* Adding posts (i.e. collocated source-receiver pairs) */
for $i \leftarrow 1$ ***to*** $\min(n_s, n_r)$ **do**
  $\Theta_\omega^{s/r} \leftarrow \{(S \cup \{e\}, R \cup \{e\}) \mid e \in E \wedge e \notin S \cap R\}$
  $\omega \leftarrow \underset{\omega' \in \Theta_\omega^{s/r}}{\arg\max} f(\omega')$
Phase 2 ; /* Adding single sensors (i.e. sources or receivers) */
for $i \leftarrow 1$ ***to*** $\max(n_s, n_r) - \min(n_s, n_r)$ **do**
  if $n_s > n_r$ **then**
    $\Theta_\omega^{s} \leftarrow \{(S \cup \{e\}, R) \mid e \in E \wedge e \notin S\}$
    $\omega \leftarrow \underset{\omega' \in \Theta_\omega^{s}}{\arg\max} f(\omega')$
  else if $n_s < n_r$ **then**
    $\Theta_\omega^{r} \leftarrow \{(S, R \cup \{e\}) \mid e \in E \wedge e \notin R\}$
    $\omega \leftarrow \underset{\omega' \in \Theta_\omega^{r}}{\arg\max} f(\omega')$
return ω

Thus, for Phase 1, given a network $\omega = (S, R) \in \Omega$ and assuming we wish to add a post on a deployment position $e \in E$, we then have

$$\omega_{s/r} \leftarrow (S \cup \{e\}, R \cup \{e\}), \tag{23}$$

$$P_d^{\omega_{s/r}}(t) = 1 - \left(\overline{P}_d^{\omega}(t) \prod_{(s,r) \in \Xi_{\omega_{s/r}} \setminus \Xi_\omega} \left(1 - P_d^{(s,r)}(t)\right) \right). \tag{24}$$

The reasoning is similar for Phase 2. The spatial complexity of this optimization is $\mathcal{O}(|T|)$ and it reduces the time complexity from $\mathcal{O}(|E||T||S||R|)$ to $\mathcal{O}(|E||T|(|S| + |R|))$ for Phase 1 and $\mathcal{O}(|E||T||S|)$ or $\mathcal{O}(|E||T||R|)$ for Phase 2, depending on the type of remaining sensors (if any). This improvement has therefore a non-negligible impact when the number of sensors becomes significant compared to the number of maritime cells.

5.2.2. Precomputations

All the collisions with obstacles are precomputed and stored in a matrix $O \in M_{|T|,|E|}(\{0, 1\})$. We then have $O_{t,e} = 1$ if there is an obstacle between the target $t \in T$ and the position $e \in E$ and $O_{t,e} = 0$ otherwise. The space complexity of this matrix is therefore $\mathcal{O}(|T||E|)$ and it is possible to use bits to store the information and thus reduce the amount of memory required. Note that there is an interesting symmetry to be exploited which can halve

the collision precomputations, i.e. $O_{t_j,e_i} = O_{t_i,e_j} \, \forall (i,j) \in [\![1,m]\!]^2, i \neq j$. However, it only works if the deployment positions coincide with the target positions, which is the case here, but not in a more general case.

5.2.3. Partial updates

It turns out to be unnecessary to update the probability of detection of a target $t \in T$ if it is already detected by the current network $\omega \in \Omega$, i.e. $P_d^\omega(t) \geq \phi$ or $1 - \overline{P}_d^\omega(t) \geq \phi$ since the probabilities of non-detection are stored in memory. In this way, we define an alternative objective function to evaluate the marginal gain at each iteration, considering only non-detected targets:

$$\begin{aligned} g \colon \Omega &\to [0,1] \\ \omega &\mapsto \frac{1}{|T|} \sum_{\substack{t \in T \\ 1-\overline{P}_d^\omega(t) < \phi}} I(P_d^\omega(t)) \,. \end{aligned} \tag{25}$$

If needed, the effective detection probabilities will be calculated at the end of the heuristic, which can be useful if one wishes to pursue with this solution (i.e. network) as a starting point for a metaheuristic, for example.

5.2.4. Symmetries

During Phase 1, there are interesting symmetries to exploit. Indeed, let us suppose that we add a pair $(s_1, r_1) \in \Xi \setminus \Xi_\omega$ to the current network $\omega \in \Omega$ and consider a pair $(s_2, r_2) \in \Xi_\omega$ (if any). We then have three newly formed sonar systems: (s_1, r_2), (s_2, r_1) and (s_1, r_1). For a given target $t \in T$, this gives us three detection probabilities to compute, but it can be noticed that $P_d^{(s_1,r_2)}(t) = P_d^{(s_2,r_1)}(t)$, which means that it is not necessary to compute this last probability. This brings us from a time complexity of $\mathcal{O}(|E||T|(|S| + |R|))$ (see 5.2.1) to a complexity of $\mathcal{O}(|E||T||S|)$ (or $\mathcal{O}(|E||T||R|)$) for Phase 1.

5.2.5. Redundant computations

For a sonar system $(s, r) \in \Xi$ and for each detection probability computation, it is necessary to calculate the following three distances: target-source $d_{t,s}$, target-receiver $d_{t,r}$ and source-receiver $d_{s,r}$.

A first option is to precalculate all distances in advance. We then have the following matrices: $D^{t \Leftrightarrow e} \in M_{|T|,|E|}(\mathbb{R}^+)$ for the distances between each target $t \in T$ and each position $e \in E$ and $D^{e \Leftrightarrow e} \in M_{|E|,|E|}(\mathbb{R}^+)$ for the distances between each pair of positions $(e, e') \in E^2$ (the latter matrix being in fact upper triangular). The space complexity associated with the addition of these matrices is $\mathcal{O}(|T||E|)$ for $D^{t \Leftrightarrow e}$ and $\mathcal{O}(|E|^2)$ for $D^{e \Leftrightarrow e}$. Even when storing floats in single or half precision, this represents a certain amount of

memory.

A second option is to store only the distances between the sensors of the current network and the targets/positions by making updates at each iteration, i.e. after having deployed one or more sensors on a given deployment position. This means that for each position $e \in E$ probed in the next iteration, we will only have to compute the distance $d_{t,e}$ for each target $t \in T$, all the other distances being already stored in memory. Note that at most $l = \max(n_s, n_r)$ deployment positions will be used throughout the heuristic, as we will deploy $\min(n_s, n_r)$ posts and $\max(n_s, n_r) - \min(n_s, n_r)$ single sensors (if the heuristic is not interrupted prior to the end). Thus we will have $D^{t \Leftrightarrow e} \in M_{|T|,l}(\mathbb{R}^+)$ and $D^{e \Leftrightarrow e} \in M_{|E|,l}(\mathbb{R}^+)$ (actually $l-1$, because we do not need to store the distances in the last iteration). The space complexity associated with the addition of these matrices is $\mathcal{O}(|T|l)$ for $D^{t \Leftrightarrow e}$ and $\mathcal{O}(|E|l)$ for $D^{e \Leftrightarrow e}$ which is more memory efficient than the first option. For the upcoming experimentations, we will therefore favor this second option, although the first one is quite feasible on architectures with large memory capacities or for small-sized instances.

All the above is valid in the most general case, i.e. where the deployment positions and the targets do not necessarily overlap, but for this particular problem that we consider here, it is possible to have only one matrix which stores the distance between the centers of each of the maritime cells (which would be upper triangular if we wanted to calculate all distances). However, this does not change the fact that the second version remains more efficient in terms of memory.

5.2.6. Maximum detection range

Given a threshold value $\epsilon \in [0, 1] \simeq 0$, a target $t \in T$ will be considered out of range of a sonar system $(s, r) \in \Xi$ if the probability of detection falls below the predefined threshold, i.e. whenever $P_d^{(s,r)}(t) < \epsilon$. This means that the target will not be considered if

$$\rho_{t,s,r} > \rho_0 \left(1 + b \log_{10}\left(\frac{1}{\epsilon} - 1\right)\right) = \rho_{max}^{\epsilon}\,, \tag{26}$$

where ρ_{max}^{ϵ} is thus the maximum range for a sonar system in monostatic configuration, here defined for the class of Fermi functions. The value of ρ_{max}^{ϵ} being pre-computed, it is then only necessary to carry out a single comparison to discard or not the target $t \in T$ and this saves us the computation of the detection probability $P_d^{(s,r)}(t)$ for a given sonar system $(s, r) \in \Xi$. This is of interest because the computation of this probability of detection is composed of multiple arithmetic operations and performed a large number of times during the heuristic. Moreover, this opens the way to

the computation of upper bounds.

5.2.7. Upper bounds

Using the maximum detection range ρ^{ϵ}_{max} defined above, it is possible to derive an upper bound, i.e. a maximum number of targets that can be reached if a post or a single sensor is placed at a given deployment position with respect to a current network.

To do this, we first introduce the matrix $D^{t\Leftrightarrow s^*} \in M_{|T|,1}(\mathbb{R}^+)$ containing the minimum distance between a target and its closest source and the matrix $D^{t\Leftrightarrow r^*} \in M_{|T|,1}(\mathbb{R}^+)$ containing the minimum distance between a target and its closest receiver. It basically boils down to obtaining a discrete Voronoi tessellation (Aurenhammer, 1991) where the generators/seeds are here the sensors already deployed and which evolves dynamically at each iteration (w.r.t. last choice made). The space complexity associated with these matrices is $\mathcal{O}(|T|)$. Indeed, it is only necessary to store the minimum distance, because for a deployment position $e \in E$, if a target $t \in T$ is not accessible with its closest sensor of the other type (source or receiver), then it will not be accessible with any other sensor (regardless of the direct blast effect).

Therefore, after each iteration and after deploying a post or a single sensor on a deployment position, these matrices are updated if the minimum distance has changed (i.e. a sensor has been placed closer to the target). Now, given a current network $\omega \in \Omega$, it is possible to define the set A^e_ω of targets accessible from position $e \in E$ if we deploy a post or a single sensor.

More precisely, for Phase 1, this gives us the following set:

$$A^e_\omega = \{t \in T \mid 1 - \overline{P}^{\omega}_d(t) < \phi \wedge X\}\,, \tag{27}$$

where

$$X = d_{t,e} \leq \frac{{\rho^{\epsilon}_{max}}^2}{D^{t\Leftrightarrow r^*}_t} \vee d_{t,e} \leq \frac{{\rho^{\epsilon}_{max}}^2}{D^{t\Leftrightarrow s^*}_t} \vee d_{t,e} \leq \rho^{\epsilon}_{max}\,. \tag{28}$$

Knowing that if for a target $t \in T$ neither of the two matrices has been updated, then the target remains inaccessible at the next iteration for the deployment position $e \in E$ under consideration. Moreover, the case where $d_{t,e} \leq \rho^{\epsilon}_{\max}$ can be precalculated upstream: it is the case where we consider the sonar system in monostatic configuration and it does not depend on the current network. It should also be noted that, since only monostatic sonar systems are added in Phase 1, it is sufficient to test only one of the first two conditions (the minimum distances being identical). For Phase 2, knowing that only one type of sensor is added iteratively, it will not be necessary to update the matrix of minimum distances of the other type of sensor (as

none will be added) and this set will therefore be precalculated to be more computationally-frugal.

5.2.8. Lower bounds

There is a lower bound that can be easily computed at each iteration and which avoids redundant computations. Let us take the example of Phase 1. At a given iteration, when probing a given position $e \in E$, it is possible that a certain number of targets $t \in T$ may be detected if we add a post on this position (unused for the moment) with respect to the current $\omega \in \Omega$ network. In the next iteration and for an unused deployment position $e \in E$, if we are still in Phase 1, then it will not be necessary to recompute the detection probabilities of the targets that we already knew were potentially detected if we had placed a pair of sensors on the considered position. Indeed, we know that these targets will be detected if we place this pair on this deployment position regardless of the last choice made, as we have already probed this position for the possible inclusion of this pair in the previous iterations (the detection probabilities can only increase).

More specifically, for a current network $\omega \in \Omega$, we then define the set of targets not yet detected by the current network but which would be detected if we added a sonar system on the position $e \in E$, giving us a temporary network $\omega_{s/r} = (S \cup \{e\}, R \cup \{e\})$:

$$C_\omega^e = \{t \in T \mid 1 - \overline{P}_d^\omega(t) < \phi \wedge P_d^{\omega_{s/r}}(t) \geq \phi\}\,. \tag{29}$$

These sets are updated at each iteration and so we have a lower bound $|C_\omega^e|$ of the number of targets that can potentially be detected if we place a pair of sensors on the position $e \in E$. This leads us to modify the previously defined objective function g as follows:

$$\begin{aligned} g\colon \Omega &\to [0,1] \\ \omega &\mapsto \frac{1}{|T|}\left(|C_\omega^e| + \sum_{\substack{t \in T \\ 1-\overline{P}_d^\omega(t) < \phi \wedge t \notin C_\omega^e}} I\left(P_d^\omega(t)\right)\right). \end{aligned} \tag{30}$$

In other words, at each iteration, we need to evaluate only the targets that are in range, not yet detected, and that we do not know yet if the inclusion of a pair on the position $e \in E$ will lead to their detection with respect to the current network and the last choice made. The reasoning is similar for Phase 2 and the sets must be reinitialized at the beginning of the phase.

5.2.9. Additional constraints using bounds

Using the previously defined upper and lower bounds, it is now possible to discard certain deployment positions in a given iteration.

First, it will be unnecessary to probe positions $e \in E$ such that $|A^e_\omega| \leq \max_{e' \in E \setminus \{e\}} |C^{e'}_\omega|$, as we can detect at most $|A^e_\omega|$ targets by deploying a post or a single sensor, while by deploying on the position with the best lower bound, we are guaranteed to detect at least $\max_{e' \in E \setminus \{e\}} |C^{e'}_\omega|$ targets. Then, for a given network $\omega \in \Omega$, suppose that we are in a given iteration and that the best candidate position is able to detect z^* targets. If we probe a position $e \in E$ not yet examined and such that $|A^e_\omega| \leq z^*$, then this position can be discarded because we will not be able to cover more targets than the current solution. By going even further, it is also possible to dismiss a position that is currently being probed by continuously re-estimating the number of targets that can be detected at most (because the targets are not sorted by increasing distance, we do not know which targets are accessible).

5.2.10. Lazy evaluations

When adding a post or a single sensor to the current network, there is no need to calculate all the detection probabilities if the target is detected before all the newly formed sonar systems have been examined. Suppose we have a current network $\omega \in \Omega$ and we want to probe the position $e \in E$ to see if adding a post on this position leads to the detection of the not yet detected target $t \in T$ during Phase 1, i.e. $1 - \overline{P}^\omega_d(t) < \phi$. We thus have a hypothetical network $\omega_{s/r} = (S \cup \{e\}, R \cup \{e\})$ and it is then sufficient to stop as soon as $P^{\omega_{s/r}}_d(t) \geq \phi$ when evaluating the newly formed sonar systems $(s, r) \in \Xi_{\omega_{s/r}} \setminus \Xi_\omega$. The reasoning is similar for Phase 2. Note that it would be conceivable to sort the deployment positions $e \in E$ by decreasing distances for each target $t \in T$ in order to terminate the evaluation sooner (on average). However, this would have a non-negligible cost in terms of complexity, as it requires sorting $|T|$ vectors of $|E|$ elements. Therefore, this has not been considered here.

5.2.11. Remarks

One idea might have been to precalculate the set of detection probabilities for each possible sonar system $(s, r) \in \Xi$ and each target $t \in T$. However, the spatial complexity of such a matrix is $\mathcal{O}(|E|^2|T|)$. Even with single or half precision floats, this does not seem like a good idea for designing an efficient heuristic, especially since there may not be enough memory available for large-scale instances. Also, it would be possible to sort the candidate deployment positions by decreasing upper bounds in order to interrupt the search more quickly. Or, similarly, to sort the positions by decreasing lower bounds in order to prioritize the most promising positions and thus potentially interrupt the search faster. Apart from these two remarks and unless

otherwise stated, all improvements have been taken into account for the experiments to follow.

6. Numerical Experiments

First and foremost, we refer to the exact resolution by the acronym OK1-S and the approximate resolutions by Greedy-N and Greedy-I for the naïve and improved versions respectively.

Concerning the experimentations, they were carried out on a Debian 10 server (64 bits) with 32 GB of RAM and 2 processors clocked at 2 Ghz (32 cores each). More specifically, the exact resolutions were performed using IBM ILOG CPLEX 20.1 (IBM, 2022) with default settings and 8 threads in parallel. In addition, we have set a maximum computational budget of 1 hour for the different resolutions and the best solution found so far is returned (if it exists). For the exact resolution, in table 2, the presence of an asterisk (*) means that it is the optimal solution and the presence of a cross (x) means that the model could not be loaded into memory due to its size. Finally, the several parameters set for these experimentations are summarized in table 1. The objective of these numerical experiments is therefore to carry out a comparison between the different resolution methods and, more specifically, between the exact resolution method (OK1-S) and the approximate resolution methods (Greedy-N and Greedy-I). To achieve this, we use the following two criteria: CPU time and coverage rate. When the maximum CPU time (1h) is reached, we focus on the best integer solution obtained (if one exists).

To conduct these experimentations, we have built up a dataset of 27 instances that were derived from DEMs of coastal-based AoIs selected throughout the globe and in such a way as to encompass a wide spectrum of peculiar geometric situations (the DEMs are made available here [1]). Indeed, to the best of our knowledge, there is no benchmark of this type that has been made accessible so far, it was therefore necessary to produce one. In particular, these DEMs (i.e. regular rectangular grids) are derived from bathymetric (depth) and topographic (height) data made publicly-available by the GEneral Bathymetric Chart of the Oceans (GEBCO) (GEBCO Bathymetric Compilation Group, 2022). The GEBCO global grid have a resolution of 15 arc-seconds, which corresponds to the dimensions of each grid cell (elevation is taken at the center point). A resolution of 15 arc-seconds corresponds to an area of about 463 square meters at the equator and this surface decreases progressively on more extreme latitudes, the circles of latitude being

[1] https://github.com/owein-thuillier/DEMs-dataset/

of smaller circumference. More precisely, to create these instances, we selected AoIs on coastlines displaying a variety of geometries and retrieved the elevation data from the GEBCO global grid (taking care also to vary the dimensions). Besides, note that the DEMs have been preprocessed to locate the isolated maritime cells (elevation less than or equal to 0) in the middle of terrestrial cells (e.g. lakes or ponds) and to assign them an arbitrary elevation so as not to take them into consideration. Without going into details, this preprocessing was done by transforming the set of maritime cells of the grid into a graph and by subsequently carrying out a computation of the connected components (He et al., 2017). For the various instances, thumbnails are available at the beginning of each row of table 2 to roughly visualize the geometry of the AoI under consideration.

Moreover, instances are ordered by increasing grid size and increasing number of maritime cells, so as to progressively increase the solving complexity. Also, the number of sources and receivers were chosen empirically in order to have a satisfactory coverage rate (i.e. more than 80%). In the following, we refer to an instance by the denomination *width-height-m* where $m \in \mathbb{N}$ is the number of maritime cells. For example, the instance 25-25-542 indicates that this is a DEM (a grid) of dimension 25x25 with 542 maritime cells (the terrestrial cells may be inferred and are 83 in total).

ρ_0	c	τ	r_b	ϕ	Fermi (b)	ϵ	σ	T_{max}
5	1.5	1	0.75	0.95	0.2	10^{-2}	0.0	3600

Table 1: Parameters set for the numerical experiments.

As shown in table 2, it is no longer possible to load the model in memory beyond instance *25-25-542* due to an excessive number of variables and constraints for the exact resolution OK1-S. Then, we manage to obtain an optimal solution up to instance *20-20-187* and non-zero integer solutions up to instance *25-25-336*. The computational budget of 1 hours becomes insufficient from instance *20-20-234* for the exact resolution OK1-S and from instance *75-75-4282* for the approximate resolution Greedy-N.

For instance *15-15-146*, Greedy-N is 20650 times faster than OK1-S and Greedy-I 282296 times faster (real CPU time: 0.004389 s) for only 0.69% less coverage. Also, for instance *20-20-187*, Greedy-N is 12452 times faster than OK1-S and Greedy-I 186790 times faster to find the optimal solution that covers 100% of the AoI. Concerning instance *20-20-234*, the computational budget having been reached for OK1-S, the best integer solution with a coverage rate of 94.87% is returned and is not optimal. Indeed, since Greedy-N and Greedy-I find a solution covering 96.15% of the AoI, the op-

timal solution is thus necessarily at least as good.

For instance *75-75-3083*, Greedy-I is 660 times faster than Greedy-N and this gap widens considerably with the size of the considered instance. In particular, for instance *175-175-26443*, an admissible solution covering 92.81% of the AoI is found by Greedy-I in 966.14 seconds, while the admissible solution obtained at the end of the one-hour computational budget by Greedy-N covers only 1.16% of the area.

In summary, we have Greedy-N and Greedy-I which enable us to find admissible solutions of high quality in a more than reasonable time compared to OK1-S. Moreover, we have the improved version of the greedy heuristic Greedy-I which outperforms the naïve version Greedy-N on larger instances. More specifically, we are able to find a solution that covers 92.81% of a 175x175 grid with 26443 maritime cells, 60 sources and 120 receivers in about 15 minutes. For instances of size 15x15, the exact resolution (OK1-S) finds better quality solutions (albeit in a much longer time) than the approximate methods (respectively Greedy-N and Greedy-I). This is expected, as when the exact method finds a solution within the allotted time budget (1 hour), it is necessarily optimal, and the greedy heuristics can do no better in terms of solution quality (i.e. coverage rate). Finally, an illustration of the solution obtained for instance *50-50-2304* with the greedy heuristic is available in Figure 6 (6 sources and 10 receivers). From left to right, we have the blank grid (DEM) of the AoI with elevation data from GEBCO (a), the heatmap with detection probabilities where the contour line (in white) corresponds to the detection threshold of 0.95 (b) and, finally, the set of covered cells, i.e. those whose target in the center is considered as detected (c).

7. Conclusions and Further Research

In this paper, we have proposed an efficient two-phase greedy heuristic to solve the Area Coverage (AC) problem in the scope of MSNs, a special case of WSNs, while taking into account existing coastlines. This is done with regard to a limited number of sensors and a given probabilistic detection model, including amongst other things a masking zone called "direct blast effect" that may occur in practice. The numerical experiments have indeed demonstrated the effectiveness of the heuristic compared to an exact model and its interest for the resolution of large-scale instances in an operational context. Besides, one of the notable advantages of such a constructive heuristic is the deterministic aspect, meaning that the same solution is found at each resolution. Moreover, the resolution of the AC problem being realized through a reduction to a Point Coverage (PC) problem, the work carried out in the scope of this study can naturally be used for the latter problem and, by extension, for the Barrier Coverage (BC) problem that can

Problem size				CPU time (seconds)			Coverage rate (%)		
Grid size	m	n_s	n_r	OK1-S	Greedy-N	Greedy-I	OK1-S	Greedy-N	Greedy-I
15 x 15									
	105	1	2	19.31	0.01	0.00	**85.71***	82.86	82.86
	125	2	2	186.72	0.04	0.00	**95.20***	94.40	94.40
	146	2	2	1239.00	0.06	0.00	**92.47***	91.78	91.78
20 x 20									
	187	2	2	1867.90	0.15	0.01	**100.00***	**100.00**	**100.00**
	234	2	2	3600.00	0.21	0.02	94.87	**96.15**	**96.15**
	291	2	2	3600.00	0.26	0.02	5.84	**84.88**	**84.88**
25 x 25									
	336	2	2	3600.00	0.40	0.03	79.76	**85.71**	**85.71**
	419	2	2	3600.00	0.78	0.06	0.00	**93.32**	**93.32**
	542	2	3	x	1.76	0.09	x	**84.50**	**84.50**
50 x 50									
	937	2	6	x	20.61	0.34	x	**87.30**	**87.30**
	1454	3	7	x	80.34	0.65	x	**83.15**	**83.15**
	2304	6	10	x	964.14	2.01	x	**83.38**	**83.38**
75 x 75									
	3083	6	12	x	3548.55	5.37	x	**82.65**	**82.65**
	4282	8	15	x	3600.00	8.89	x	50.65	**83.05**
	5343	10	18	x	3600.00	15.49	x	38.03	**81.08**
100 x 100									
	6360	12	24	x	3600.00	20.36	x	21.53	**81.54**
	7524	14	28	x	3600.00	40.61	x	18.45	**85.63**
	8947	16	32	x	3600.00	51.34	x	15.25	**80.46**
125 x 125									
	10490	18	36	x	3600.00	98.93	x	8.16	**91.82**
	12224	22	42	x	3600.00	103.77	x	8.78	**82.51**
	14239	26	52	x	3600.00	179.60	x	4.68	**83.16**
150 x 150									
	15525	28	56	x	3600.00	237.41	x	6.39	**97.40**
	17026	30	60	x	3600.00	389.45	x	5.04	**94.86**
	18948	37	74	x	3600.00	408.79	x	3.30	**82.77**
175 x 175									
	20675	40	80	x	3600.00	652.97	x	1.76	**92.82**
	24179	50	100	x	3600.00	876.51	x	1.35	**91.83**
	26443	60	120	x	3600.00	966.14	x	1.16	**92.79**

Table 2: Experimental results on the 27 instances.

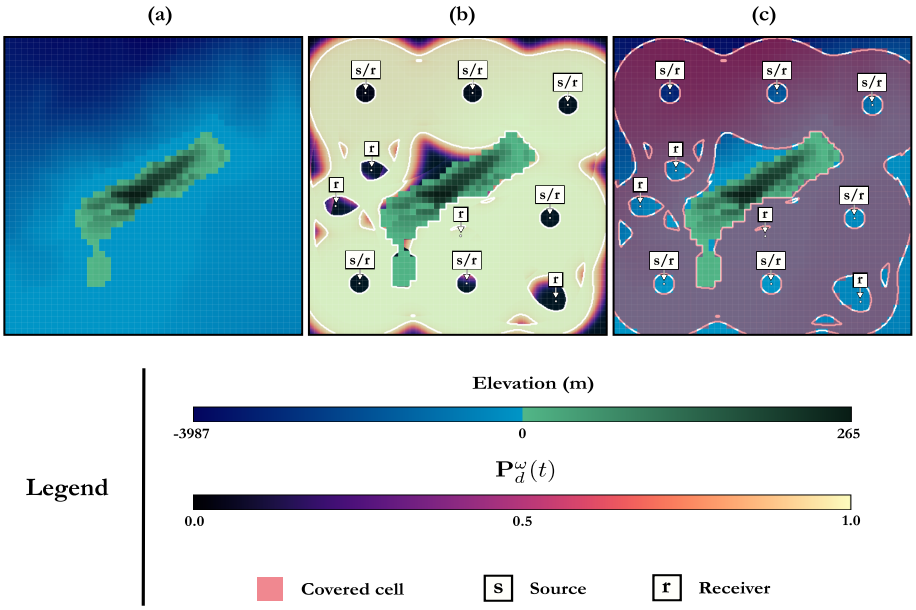

Figure 6: Solution obtained after solving instance *50-50-2304* with the greedy heuristic.

be discretized in the same way. It is also noteworthy that this heuristic can be easily adapted to address the dual problem of minimizing the number of sensors to cover the entire AoI with costs associated to the deployment of sources and receivers (full coverage variant of the AC problem). Besides, the heuristic is generic enough to allow the use of more accurate acoustic models, such as the use of ray-tracing to estimate the performances of different sonars (it is then sufficient to make a 2D projection and to compute the detection probabilities according to the transmission losses at all points of space). Finally, the work undertaken here can be repurposed to tackle coverage problems in Multistatic Radar Networks (MRNs), which are mostly based on similar mathematical concepts.

Until now, there has been no competing approximate resolution method for the problem at hand, making it impossible to provide any comparison. Therefore, we proposed the first constructive heuristic (greedy-based) in order to generate high-quality solutions in a reasonable amount of time. In future work, these solutions could be used in dedicated metaheuristics to derive higher quality solutions (e.g. Local Search (LS), Variable Neighborhood Search (VNS), Variable Neighborhood Descent (VND), Tabu Search (TS) or Simulated Annealing (SA)). Another possibility would be to handle the problem using a sector-based approach by encompassing a set of deployment positions. In this way, it would be possible to discard the less interesting sectors by means of appropriate lower and upper bounds, thus reducing the time required per iteration. To another extent, a possible way of exploration

could be the decomposition into sub-problems in order to use methods such as benders decomposition, Lagrangian relaxations or even columns generation. Lastly, a refinement of the detection model could be considered for a greater degree of realism from an operational standpoint by reinstating, for example, absorption losses that have been largely neglected in the literature related to optimal deployment in MSNs. Note that if one wishes to incorporate a more realistic performance model, care should be taken to vary the instances and build a more comprehensive benchmark so as to have different types of bottom depths (shallow, deep) and not just coastal-based instances, most of which have shallow waters.

References

Abbot, P. and Dyer, I. (2002). *Sonar Performance Predictions Incorporating Environmental Variability*. Springer Netherlands, Dordrecht.

Ainslie, M. (2010). *Principles of Sonar Performance Modelling*. Springer, Berlin, Heidelberg.

Amanatides, J. and Woo, A. (1987). A fast voxel traversal algorithm for ray tracing. *Proceedings of EuroGraphics*, 87.

Amanipour, V. and Olfat, A. (2011). CFAR detection for multistatic radar. *Signal Processing*, 91:28–37.

Aurenhammer, F. (1991). Voronoi diagrams—a survey of a fundamental geometric data structure. *ACM Comput. Surv.*, 23(3):345–405.

Been, R., Jespers, S., Coraluppi, S., Carthel, C., Strode, C., and Vermeij, A. (2007). Multistatic sonar: a road to a maritime network enabled capability. NATO Undersea Research Centre (NURC).

Chaovalitwongse, W., Pardalos, P. M., and Prokopyev, O. A. (2004). A new linearization technique for multi-quadratic 0–1 programming problems. *Operations Research Letters*, 32(6):517–522.

Church, R. L. and Revelle, C. S. (1974). The maximal covering location problem. *Papers of the Regional Science Association*, 32:101–118.

Cox, A. W. (1974). *Sonar and underwater sound*. Lexington Books, Lexington, Mass.

Cox, H. (1989). *Fundamentals of Bistatic Active sonar*. Springer, Dordrecht.

Craparo, E. M., Fügenschuh, A., Hof, C., and Karatas, M. (2019). Optimizing source and receiver placement in multistatic sonar networks to monitor fixed targets. *European Journal of Operational Research*, 272(3):816–831.

Craparo, E. M. and Karatas, M. (2018). A method for placing sources in multistatic sonar networks. Technical Report NPS-OR-18-001, Naval Postgraduate School, Monterey, California.

Craparo, E. M. and Karatas, M. (2020). Optimal source placement for point coverage in active multistatic sonar networks. *Naval Research Logistics*, 67(1):63–74.

Craparo, E. M., Karatas, M., and Kuhn, T. U. (2017). Sensor placement in active multistatic sonar networks: Active Multistatic Sonar Networks. *Naval Research Logistics*, 64(4):287–304.

DelBalzo, D. R. and Stangl, K. C. (2009). Design and performance of irregular sonobuoy patterns in complicated environments. In *OCEANS 2009*, pages 1–4, Biloxi, MS. IEEE.

Elhabyan, R., Shi, W., and St-Hilaire, M. (2019). Coverage protocols for wireless sensor networks: Review and future directions. *Journal of Communications and Networks*, 21(1):45–60.

Fewell, M. and Ozols, S. (2011). Simple detection-performance analysis of multistatic sonar for anti-submarine warfare. Technical Report DSTO-TR-2562, Defence Science and Technology Organisation, Edinburgh, South Australia.

Fewell, M. and Thredgold, J. (2009). Cumulative track-initiation probability as a basis for assessing sonar-system performance in anti-submarine warfare. Technical Report DSTO-TN-0932, Defence Science and Technology Organisation, Edinburgh, South Australia.

Fewell, M., Thredgold, J., and Kershaw, D. (2008). Benefits of sharing detections for networked track initiation in anti-submarine warfare. Technical Report DSTO-TR-2086, Defence Science and Technology Organisation, Edinburgh, South Australia.

Fügenschuh, A. R., Craparo, E. M., Karatas, M., and Buttrey, S. E. (2020). Solving multistatic sonar location problems with mixed-integer programming. *Optimization and Engineering*, 21(1):273–303.

GEBCO Bathymetric Compilation Group (2022). The GEBCO 2022 grid - a continuous terrain model of the global oceans and land. NERC EDS British Oceanographic Data Centre NOC.

Gong, X., Zhang, J., Cochran, D., and Xing, K. (2013). Barrier coverage in bistatic radar sensor networks: cassini oval sensing and optimal placement. In *Proceedings of the fourteenth ACM international symposium on Mobile ad hoc networking and computing - MobiHoc '13*, page 49, Bangalore, India. ACM Press.

Grasso, R., Cococcioni, M., Mourre, B., Osler, J., and Chiggiato, J. (2013). A decision support system for optimal deployment of sonobuoy networks based on sea current forecasts and multi-objective evolutionary optimization. *Expert Systems with Applications*, 40(10):3886–3899.

Guth, P., Niekerk, A., Grohmann, C., Muller, J.-P., Hawker, L., Florinsky, I., Gesch, D., Reuter, H., Herrera, V., Riazanoff, S., Lopez-Vazquez, C., Carabajal, C., Albinet, C., and Strobl, P. (2021). Digital elevation models: Terminology and definitions. *Remote Sensing*, 13.

He, L., Ren, X., Gao, Q., Zhao, X., Yao, B., and Chao, Y. (2017). The connected-component labeling problem: A review of state-of-the-art algorithms. *Pattern Recognition*, 70:25–43.

Hervé, C. (2011). *Imagerie pour le sonar à ouverture synthétique multistatique (Imaging for multistatic synthetic aperture sonar)*. PhD thesis, Aix-Marseille 1.

Hof, C. (2015). Optimization of source and receiver placement in multistatic sonar environments. Master's thesis, Naval Postgraduate School, Monterey, California.

Holler, R., Horbach, A., McEachern, J., and of Naval Research, U. S. O. (2008). *The Ears of Air ASW: A History of U.S. Navy Sonobuoys*. Navmar Applied Sciences Corporation, Warminster, Pennsylvania.

IBM (2022). IBM ILOG CPLEX Optimizer.

Iqbal, K., Zhang, M., Piao, S., and Ge, H. (2020). Evolution of sonobuoy through history & its applications: A survey. In *2020 17th International Bhurban Conference on Applied Sciences and Technology (IBCAST)*, pages 543–554.

Jensen, F. B., Kuperman, W. A., Porter, M. B., and Schmidt, H. (2011). *Computational Ocean Acoustics*. Springer New York, New York, NY.

Karatas, M. (2013). A multi foci closed curve: Cassini oval, its properties and applications. *Doğuş Üniversitesi Dergisi*, 2:231–248.

Karatas, M. and Craparo, E. M. (2015). Evaluating the direct blast effect in multistatic sonar networks using Monte Carlo simulation. In *2015 Winter Simulation Conference (WSC)*, pages 1184–1194, Huntington Beach, California, USA. IEEE.

Karatas, M., Gunal, M. M., and Craparo, E. M. (2016). Performance Evaluation Of Mobile Multistatic Search Operations With Simulation. In *49th Annual Simulation Symposium (ANSS 2016)*. Society for Modeling and Simulation International (SCS).

Khoufi, I., Minet, P., Laouiti, A., and Mahfoudh, S. (2017). Survey of deployment algorithms in wireless sensor networks: Coverage and connectivity issues and challenges. *International Journal of Autonomous and Adaptive Communications Systems*, 10:341.

Kuhn, T. (2014). Optimal sensor placement in active multistatic sonar networks. Master's thesis, Naval Postgraduate School, Monterey, California.

Li, H.-P., Feng, D.-Z., Chen, S.-F., and Zhou, Y.-P. (2021). Deployment Optimization Method of Multistatic Radar for Constructing Circular Barrier Coverage. *Sensors*, 21(19):6573.

Li, H.-P., Feng, D.-Z., Liu, C., and Zhang, C. (2022). Optimal deployment of multistatic radar for belt barrier coverage. *Wireless Networks*, 28(5):2213–2235.

Lurton, X. (2002). *An introduction to underwater acoustics: principles and applications*. Springer, London ; New York.

McDougall, J. and Stoner, E. C. (1938). The computation of Fermi-Dirac functions. *Philosophical Transactions of the Royal Society of London. Series A, Mathematical and Physical Sciences*, 237(773):67–104.

Megiddo, N., Zemel, E., and Hakimi, S. L. (1983). The maximum coverage location problem. *Siam Journal on Algebraic and Discrete Methods*, 4:253–261.

Mohamed, R. E., Saleh, A. I., Abdelrazzak, M., and Samra, A. S. (2018). Survey on Wireless Sensor Network Applications and Energy Efficient Routing Protocols. *Wireless Personal Communications*, 101(2):1019–1055.

Ngatchou, P., Fox, W., and El-Sharkawi, M. (2006). Multiobjective Multistatic Sonar Sensor Placement. In *2006 IEEE International Conference on Evolutionary Computation*, pages 2713–2719, Vancouver, BC, Canada. IEEE.

Oral, M. and Kettani, O. (1992). A Linearization Procedure for Quadratic and Cubic Mixed-Integer Problems. *Operations Research*, 40(1-supplement-1):S109–S116.

Ozols, S. and Fewell, M. (2011). On the design of multistatic sonobuoy fields for area search. Technical Report DSTO-TR-2563, Defence Science and Technology Organisation, Edinburgh, South Australia.

Prim, R. C. (1957). Shortest Connection Networks And Some Generalizations. *Bell System Technical Journal*, 36(6):1389–1401.

Strode, C., Mourre, B., and Rixen, M. (2012). Decision support using the Multistatic Tactical Planning Aid (MSTPA). *Ocean Dynamics*, 62(1):161–175.

Tripathi, A., Gupta, H. P., Dutta, T., Mishra, R., Shukla, K. K., and Jit, S. (2018). Coverage and Connectivity in WSNs: A Survey, Research Issues and Challenges. *IEEE Access*, 6:26971–26992.

Urick, R. J. (1983). *Principles of underwater sound*. McGraw-Hill, New York, 3rd ed edition.

Wang, B. (2011). Coverage problems in sensor networks: A survey. *ACM Computing Surveys*, 43(4):1–53.

Washburn, A. R. (2010). A multistatic sonobuoy theory. Technical Report NPS-OR-10-005, Naval Postgraduate School, Monterey, California.

Washburn, A. R. and Karatas, M. (2015). Multistatic search theory. *Military Operations Research*, 20:21–38.

Appendix A. Improved version: complete pseudo-code

Before giving the whole pseudo-code, some general remarks about it:

- We have chosen to use breaks when necessary to make it lighter.
- Checkpoints to ensure that the computation budget is not exceeded have not been included, but they can be added where necessary (preferably deep in the nested loops).

Algorithm 2: Global pre-processing

$O \in M_{|T|,|E|}(\{0,1\})$ /* Matrix storing collisions */
if *number of terrestrial cells ¿ 0* **then**
  for $t \in T$ **do**
    for $e \in E$ **do**
      if $id(t) \leq id(e)$ **then**
        /* id() returns the index of the cell on which the target or position is located */
        $t' \leftarrow e$; $e' \leftarrow t$ /* Positions are reversed (symmetrical matrix) */
        $bool \leftarrow 0$
        if *there is an obstacle between* t *and* e **then**
          $bool \leftarrow 1$
        $O_{t,e} \leftarrow bool$
        $O_{t',e'} \leftarrow bool$
$S \leftarrow \emptyset$; $R \leftarrow \emptyset$; $\omega \leftarrow (S, R)$ /* Current network */
$z^{\omega} \leftarrow 0$ /* Number of targets covered by the current network */
$A_{\omega} \leftarrow \emptyset$; $C_{\omega} \leftarrow \emptyset$ /* Initializing sets */
[Phase 1] Pre-processing

Algorithm 3: [Phase 1] Pre-processing

for $t \in T$ **do**
  for $e \in E$ **do**
    $d_{t,e} \leftarrow compute\_distance(t, e)$
    if $d_{t,e} \leq \rho^{e}_{max}$ **then**
      /* Target in range if a post is deployed on e */
      $A^{e}_{\omega} \leftarrow A^{e}_{\omega} \cup \{t\}$
[Phase 1] Main loop

Algorithm 4: [Phase 1] Main loop

```
for i ← 1 to min(n_s, n_r) do
    /* Deployment of posts (collocated source-receiver pairs) */
    lB* ← max_{e∈E}{|C^e_ω|}                                  /* Best lower bound */
    e* ← ∅                                                    /* Best position */
    z* ← 0                                                    /* Best marginal gain */
    for e ∈ E do
        uB ← |A^e_ω|                          /* Upper bound if post deployed on e */
        if e ∉ S ∩ R and max{z*, lB*} < uB* then
            /* Position e is available and we can expect to do better */
            z ← |C^e_ω|                       /* Targets we knew were already covered */
            uB ← uB − z
            for t ∈ T do
                if z + uB ≤ z* then
                    /* It will be impossible to do better (dynamic updating) */
                    break
                if t ∈ A^e_ω and t ∉ C^e_ω then
                    /* Target accessible and to be evaluated */
                    d_{t,e} ← compute_distance(t, e)
                    P̄ ← P̄^ω_d(t)   /* Cumulative probability of non-detection */
                    P̄ ← P̄ · (1 − P^{(e,e)}_d(t))          /* Monostatic contribution */
                    if 1 − P̄ ≥ φ then
                        /* Target covered */
                        z ← z + 1
                        C^e_ω ← C^e_ω ∪ {t}   /* Information for the next iteration */
                    else
                        for e′ ∈ S ∩ R do
                            /* Contributions with existing posts */
                            if O_{t,e′} = 0 then
                                /* No obstacles between e′ and t */
                                d_{t,e′} ← D^{t⇔e}_{t,e′}                    /* In memory */
                                d_{e,e′} ← D^{e⇔e}_{e,e′}                    /* In memory */
                                if d_{t,e} ≤ ρ^e_max / d_{t,e′}^2 then
                                    /* Target in range for the considered pair */
                                    P̄ ← P̄ · (1 − P^{(e,e′)}_d(t))^2          /* Symmetry */
                                    if 1 − P̄ ≥ φ then
                                        /* Target covered */
                                        z ← z + 1
                                        C^e_ω ← C^e_ω ∪ {t}
                                        break                                /* Lazy evaluations */
                    uB ← uB − 1
        if z > z* then
            /* Better position found */
            e* ← e; z* ← z
    [Phase 1] Update
[Phase 2] Pre-Processing
```

Algorithm 5: [Phase 1] Update

if $e^* = \emptyset$ **then**
  /* No improving position */
  break
else if $z^* < \sigma$ **then**
  /* Marginal gain below the threshold */
  break
else
  /* An optimal position has been identified */
  $z^{\omega} \leftarrow z^{\omega} + z^*$ /* Update objective function (i.e. number of targets covered) */
  for $t \in T$ **do**
    if $1 - \overline{P}_d^{\omega}(t) < \phi$ ***and*** $O_{t,e^*} = 0$ **then**
      /* If the target is still not covered and accessible from the new position (no obstacle) $\Longrightarrow$ if the target is already covered, there is no need to update its (non-)detection probability */
      $d_{t,e^*} \leftarrow compute\_distance(t, e^*)$
      $bool\_update \leftarrow 0$
      if $d_{t,e^*} < D_t^{t \Leftrightarrow s^*}$ **then**
        /* This post is the new sensor closest to the t target (source and receiver) */
        $D_t^{t \Leftrightarrow s^*} \leftarrow d_{t,e^*}$
        $D_t^{t \Leftrightarrow r^*} \leftarrow d_{t,e^*}$
        $bool\_update \leftarrow 1$ /* If the minimum distance is updated, the target may become accessible from certain positions */
      $D_{t,e^*}^{t \Leftrightarrow e} \leftarrow d_{t,e^*}$
      $\overline{P}_d^{\omega}(t) \leftarrow \overline{P}_d^{\omega}(t) \cdot (1 - P_d^{(e^*,e^*)}(t))$ /* Monostatic contribution */
      if $1 - \overline{P}_d^{\omega}(t) < \phi$ **then**
        /* Target still not covered */
        for $e' \in S \cap R$ **do**
          /* Contributions with existing posts */
          if $O_{t,e'} = 0$ **then**
            /* No obstacles between e' and t */
            $d_{t,e'} \leftarrow D_{t,e'}^{t \Leftrightarrow e}$
            $d_{e^*,e'} \leftarrow D_{e^*,e'}^{e \Leftrightarrow e}$
            if $d_{t,e^*} \leq \frac{\rho_{max}^{\epsilon}}{d_{t,e'}}^2$ **then**
              /* Target in range for the considered pair */
              $\overline{P}_d^{\omega}(t) \leftarrow \overline{P}_d^{\omega}(t) \cdot (1 - P_d^{(e^*,e')}(t))^2$ /* Symmetry */
              if $1 - \overline{P}_d^{\omega}(t) \geq \phi$ **then**
                /* Target covered */
                break /* Lazy evaluations */
      [Phase 1] Update sets
  for $e \in E$ **do**
    /* Update distances from the new position added to the network to all other positions */
    $d_{e,e^*} \leftarrow compute\_distance(e, e^*)$
    $D_{e,e^*}^{e \Leftrightarrow e} \leftarrow d_{e,e^*}$
  $S \leftarrow S \cup \{e^*\}; R \leftarrow R \cup \{e^*\}$

Algorithm 6: [Phase 1] Update sets

if $1 - \overline{P}_d^{\omega}(t) \geq \phi$ **then**
  /* Target just covered: remove it from sets */
  for $e \in E$ **do**
    if $t \in A_\omega^e$ **then**
      $A_\omega^e \leftarrow A_\omega^e \; n \; \{t\}$
    if $t \in C_\omega^e$ **then**
      $C_\omega^e \leftarrow C_\omega^e \; n \; \{t\}$
else
  /* Target still not covered: it may become accessible from new positions if an update of the minimum distance has occurred */
  for $e \in E$ **do**
    if $bool\_update = 1$ ***and*** $t \notin A_\omega^e$ ***and*** $O_{t,e} = 0$ **then**
      /* There was an update and t was not accessible and there is no obstacle between t and e */
      $d_{t,e} \leftarrow compute\_distance(t, e)$
      if $d_{t,e} \leq \frac{{\rho_{max}^{\epsilon}}^2}{d_{t,e^*}}$ **then**
        /* Target t is now accessible from position e */
        $A_\omega^e \leftarrow A_\omega^e \cup \{t\}$

Algorithm 7: [Phase 2] Pre-processing

$A_\omega \leftarrow \emptyset$; $C_\omega \leftarrow \emptyset$ /* Reinitialisation of the sets */
for $t \in T$ **do**
  if $1 - \overline{P}_d^{\omega}(t) < \phi$ **then**
    /* Target not covered */
    for $e \in E$ **do**
      if $e \notin S \cap R$ ***and*** $O_{t,e} = 0$ **then**
        /* Position available and no obstacles between t and e */
        $d_{t,e} \leftarrow compute\_distance(t, e)$
        if $d_{t,e} \leq \frac{{\rho_{max}^{\epsilon}}^2}{D_t^{t \Leftrightarrow s^*}}$ **then**
          /* Target in range if a post is deployed on e */
          $A_\omega^e \leftarrow A_\omega^e \cup \{t\}$
[Phase 2] Main loop

Algorithm 8: [Phase 2] Main loop

```
for i ← 1 to max(n_s, n_r) − min(n_s, n_r) do
    /* Deployment of the remaining sensors (sources or receivers) */
    lB* ← max_{e∈E}{|C^e_ω|}                                   /* Best lower bound */
    e* ← ∅                                                      /* Best position */
    z* ← 0                                                      /* Best marginal gain */
    for e ∈ E do
        uB ← |A^e_ω|  /* Upper bound if sensor (source or receiver) deployed on
          e */
        if e ∉ S ∪ R and max{z*, lB*} < uB* then
            /* Position e is available and we can expect to do better */
            z ← |C^e_ω|                                         /* Targets we knew were already covered */
            uB ← uB − z
            for t ∈ T do
                if z + uB ≤ z* then
                    /* It will be impossible to do better (dynamic updating)
                       */
                    break
                if t ∈ A^e_ω and t ∉ C^e_ω then
                    /* Target accessible and to be evaluated */
                    d_{t,e} ← compute_distance(t, e)
                    P̄ ← P̄^ω_d(t)  /* Cumulative probability of non-detection */
                    for e′ ∈ S ∩ R do
                        /* Contributions with existing posts */
                        if O_{t,e′} = 0 then
                            /* No obstacles between e′ and t */
                            d_{t,e′} ← D^{t⇔e}_{t,e′}                    /* In memory */
                            d_{e,e′} ← D^{e⇔e}_{e,e′}                    /* In memory */
                            if d_{t,e} ≤ (ρ^e_max / d_{t,e′})^2 then
                                /* Target in range for the considered pair */
                                P̄ ← P̄ · (1 − P_d^{(e,e′)}(t))
                                if 1 − P̄ ≥ φ then
                                    /* Target covered */
                                    z ← z + 1
                                    C^e_ω ← C^e_ω ∪ {t}
                                    break                       /* Lazy evaluations */
                    uB ← uB − 1
        if z > z* then
            /* Better position found */
            e* ← e; z* ← z
    [Phase 2] Update
return ω, z^ω                                                   /* End of algorithm */
```

Algorithm 9: [Phase 2] Update

```
1  if e* = ∅ then
       /* No improving position */
2      break
3  else if z* < σ then
       /* Marginal gain below the threshold */
4      break
5  else
       /* An optimal position has been identified */
6      z^ω ← z^ω + z*        /* Update objective function (i.e. number of targets
        covered) */
7      for t ∈ T do
8          if 1 − P̄_d^ω(t) < φ and O_{t,e*} = 0 then
               /* If the target is still not covered and accessible from the
                  new position (no obstacle) ⟹ if the target is already
                  covered, there is no need to update its (non-)detection
                  probability */
9              d_{t,e*} ← compute_distance(t, e*)
10             for e' ∈ S ∩ R do
                   /* Contributions with existing posts */
11                 if O_{t,e'} = 0 then
                       /* No obstacles between e' and t */
12                     d_{t,e'} ← D^{t⇔e}_{t,e'}
13                     d_{e*,e'} ← D^{e⇔e}_{e*,e'}
14                     if d_{t,e*} ≤ (ρ^ε_max / d_{t,e'})^2 then
                           /* Target in range for the considered pair */
15                         P̄_d^ω(t) ← P̄_d^ω(t) · (1 − P_d^{(e*,e')}(t))
16                         if 1 − P̄_d^ω(t) ≥ φ then
                               /* Target covered */
17                             break                    /* Lazy evaluations */
18             if 1 − P̄_d^ω(t) ≥ φ then
                   /* Target just covered: remove it from sets */
19                 for e ∈ E do
20                     if t ∈ A^e_ω then
21                         A^e_ω ← A^e_ω n {t}
22                     if t ∈ C^e_ω then
23                         C^e_ω ← C^e_ω n {t}
24     if n_s > n_r then
           /* We add sources */
25         S ← S ∪ {e*}
26     else
           /* We add receivers */
27         R ← R ∪ {e*}
```