Sensor Placement in Active Multistatic Sonar Networks

Emily M. Craparo,[1] **Mumtaz Karatas**,[2] **Tobias U. Kuhn**[3]

[1]*Operations Research Department, Naval Postgraduate School, Monterey, California*

[2]*Department of Industrial Engineering, Turkish Naval Academy, Tuzla, Istanbul 34940, Turkey*

[3]*Operations Research Analyst, 10th Armored Division, German Army, Veitshöchheim, Germany*

Received 7 December 2016; revised 24 February 2017; accepted 8 June 2017
DOI 10.1002/nav.21746
Published online 28 June 2017 in Wiley Online Library (wileyonlinelibrary.com).

Abstract: The idea of deploying noncollocated sources and receivers in multistatic sonar networks (MSNs) has emerged as a promising area of opportunity in sonar systems. This article is one of the first to address point coverage problems in MSNs, where a number of points of interest have to be monitored in order to protect them from hostile underwater assets. We consider discrete "definite range" sensors as well as various diffuse sensor models. We make several new contributions. By showing that the convex hull spanned by the targets is guaranteed to contain optimal sensor positions, we are able to limit the solution space. Under a definite range sensor model, we are able to exclude even more suboptimal solutions. We then formulate a nonlinear program and an integer nonlinear program to express the sensor placement problem. To address the nonconvex single-source placement problem, we develop the Divide Best Sector (DiBS) algorithm, which quickly provides an optimal source position assuming fixed receivers. Starting with a basic implementation of DiBS, we show how incorporating advanced sector splitting methods and termination conditions further improve the algorithm. We also discuss two ways to use DiBS to find multiple source positions by placing sensors iteratively or simultaneously. © 2017 Wiley Periodicals, Inc. Naval Research Logistics 64: 287–304, 2017

Keywords: undersea sensing; multistatic sonar; point coverage; sensor network optimization

1. INTRODUCTION

Active sonar systems have long constituted an important sensing mechanism aboard submarines and ships in anti-submarine warfare. In a typical sonar system, a ping is sent out and the echo yields information about other objects in the area. Recently, however, the idea of deploying noncollocated sources and receivers has emerged as a promising area of opportunity in sonar systems. Despite extensive use in practical applications, there are relatively few formal results in the literature to guide deployment of such systems.

A multistatic sonar network (MSN) consisting of a number of noncollocated sources and receivers carries a number of advantages. Reference [11] states that "countermeasure tactics are greatly complicated if the target does not know the position of the receivers." This is justified by the fact that receivers do not send out pings and thus do not reveal their locations. Beyond that, recent procurement records of the United States Navy (USN) demonstrate that sources can cost up to seven times as much as receivers [41]. Thus, deploying more receivers than sources might significantly reduce costs without sacrificing performance. Also, a "multistatic system can employ different platforms for sources and receivers. A ship might be the source, while the receivers are sonobuoys" [43]; such flexibility can greatly aid deployment efforts in practice. On top of this, a ping might be received by multiple receivers, thus improving accuracy. References [9] and [38] discuss how to merge multiple detections into a single alert that is more precise and eliminates some of the false alarms that occur on traditional (monostatic) sonar systems.

However, these benefits come at a cost. The performance of a bistatic sonar system, that is, an MSN with exactly one source and one receiver, is significantly more difficult to model than a monostatic sonar system, leading to challenges in optimal deployment and usage. The primary reason for these challenges lies in the differences of the geometry of both systems. In a monostatic sonar system, the detection probability is mainly related to the distance between the sonar device and the potential target. This relationship

Correspondence to: Emily M. Craparo (emcrapar@nps.edu)

© 2017 Wiley Periodicals, Inc.

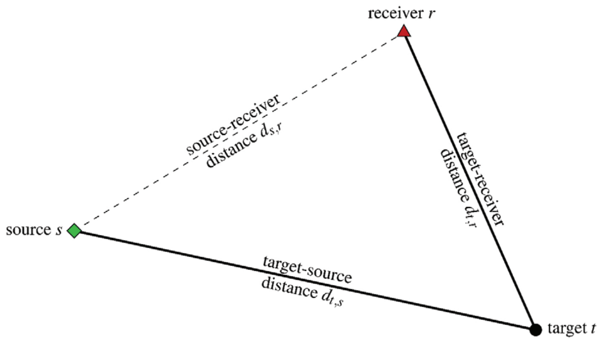
receiver r
source-receiver distance $d_{s,r}$
target-receiver distance $d_{t,r}$
source s
target-source distance $d_{t,s}$
target t

Figure 1. Geometry of a bistatic sonar system—a bistatic sonar system is composed of a noncollocated source and receiver pair. The detection probability for target t depends on both target-source and target-receiver distances. [Color figure can be viewed at wileyonlinelibrary.com.]

is more complicated in a bistatic model, where the detection probability depends on the product of the target-source and target-receiver distances. Moreover, the analytical challenges are exacerbated by MSNs involving multiple sources and receivers. The geometry of an example bistatic sonar system and the source-target-receiver distances are displayed in Fig. 1.

Three main types of problems occur in the sonar literature: barrier search, area search, and point coverage. This article is one of the first to focus on the point coverage problem, in which the goal is to place a finite number of sensors (sources and receivers) so as to best cover a set of stationary points of interest (PoIs). These PoIs may correspond to strategic friendly locations that must be kept under surveillance, such as the vicinity surrounding oil platforms, aircraft carriers, ports, or other high value assets; or they may simply represent a discretization of a barrier or area coverage problem. If the goal is to detect an intruding enemy, we may have some prior knowledge of where the enemy is likely to be found, or there may be greater consequences for failing to detect the enemy in some locations than in others. In such cases, it is straightforward to associate with each PoI a weight reflecting its relative importance. To be in accordance with the terminology of other literature, we will refer to PoIs as "targets," and instead of "covering" these targets, we "detect" them. However, we emphasize that in this article, a "target" is a *location in space that we wish to monitor*.

In the course of this study, we use two sensor models. The first is a simple definite range sensing model, sometimes called a "cookie cutter" sensor. In this sensor model, a target is detected with probability 1 if it is within the detection range of the sensor network, and otherwise it is not detected. We also utilize more sophisticated sensor models in which the probability of detection varies between zero and one depending on the configuration of the sensor network and the location of the target. Much of this work was documented in the master's thesis of the third author [31]; additional details and computational experiments appear there.

This article is organized as follows. In Section 2, we review a selection of prior work relating to multistatic systems. Section 3 describes our assumptions and the aspects of MSN performance modeling relevant to our work. Our new contributions appear in Sections 4 and 5. In Section 4, we develop some general observations about point coverage sensing with MSNs. We prove that a convex hull encasing the targets contains optimal sensor locations. Furthermore, we show the importance of various geometric constructs in definite range sensor models. We show that these constructs can be used to derive performance bounds for MSNs. In Section 5, we first formulate a nonconvex nonlinear program (NLP) and a nonconvex integer nonlinear program (INLP) to express the sensor placement problem. Next, we focus on the development and enhancement of the Divide Best Sector (DiBS) algorithm to find the optimal position for a single source, assuming fixed targets and receivers in place. This algorithm takes advantage of problem structure to find a globally optimal position for the source without solving a nonconvex optimization problem. We investigate many details of the algorithm and assess methods and means to improve it. Finally, Section 6 contains a summary of our findings.

2. LITERATURE REVIEW

Sonar system research belongs to a wide field with diverse subcategories. There are a number of studies in the literature that consider the problems of multistatic performance prediction and sensor placement. For example, Refs. [17] and [40] apply a genetic algorithm to determine the locations of multistatic sensors for maximizing area coverage. Reference [34] uses a particle swarm optimization technique for the same objective. Reference [39] utilizes game theory to select positions for multistatic sensors with the goal of detecting transiting intelligent agents. Reference [23] uses a simulation-based methodology for multistatic search and rescue missions. Reference [7] studies the connectivity problem in a mobile multistatic radar network containing unmanned air vehicles. They develop a metric that provides for a balance between the performance and connectivity of the network. Reference [18] studies the barrier coverage problem, in which sensors are deployed on a line segment. They determine the optimal placement order and spacing of sensors which minimizes the vulnerability of the network against intruders. Reference [21] analyzes the performance of a MSN by using a combined Monte Carlo simulation and Bayesian integration technique. They use this methodology to account for uncertainties such as enemy behavior and probable locations. Reference [4] develops a multistatic performance prediction methodology which can be used to assess the detection performance of a MSN as a function of source and receiver densities. Reference [42] computes the expected detection probability of a given track in a MSN field where all sources and receivers are distributed uniformly at random.

The Australian Defence Science and Technology Organisation (DSTO) analyzes multiple scenarios. Reference [14] compares a field of monostatic sonar systems with that of a field of similar sonars operated multistatically, where sources and receivers are collocated. The direct comparison of the two modes of operation reveals that the correct choice of sensor models affects the outcome. Using a definite range sensor model, the researchers find no advantage to the MSN. However, using the exponential model, the researchers achieve comparable sensing capabilities in an MSN with about one quarter the number of sensors as required in a monostatic setup. This finding is important since it reduces the number of pings in a given field. As soon as a ping is sent out, a hostile submarine knows the location of the source and will depart, which consequently makes it harder to detect. Another reason to reduce the number of pings is the artificial stress for sea dwellers produced by sonar systems. In another study, Ref. [35] studies the area coverage problem and analyzes the coverage performance of 27 MSN layouts to determine the most cost-effective pattern.

There are different approaches to quantify the effectiveness of the deployment and usage of multistatic sensors. A strategic rather than a tactical approach is analyzed in Ref. [44]. The authors consider a randomly deployed MSN and develop an analytic theory that measures the coverage of the network as a function of source and receiver densities. The most interesting aspect about this approach is the fact that it does not need to consider the geometric arrangement of the sensors. Using some analytical results from Ref. [44], studies Refs. [25] and [26] use Monte Carlo simulation to test the effect of the direct blast zone and mobile searchers on the performance of a MSN, respectively.

The aforementioned studies only consider the area coverage, barrier coverage, and tracking performance of a MSN. In contrast, Ref. [13] engages the point coverage problem for MSNs. The authors assume fixed PoIs and receivers and discuss various approaches to optimally place multiple sources.

3. ASSUMPTIONS AND PRELIMINARIES

3.1. Assumptions

We consider a set of sources S, a set of receivers R, and a set of targets T. Each target is associated with a point in the two-dimensional Euclidean plane with homogeneous environmental conditions. We consider the problem of placing both types of sensor (sources and receivers), as well as the special case in which receivers are already deployed and we only wish to select locations for the sources. The direct blast zone is the area where targets cannot be detected because their echoes arrive at nearly the same time as the ping from the source (see Refs. [11, 14], and [25] for details). This effect can be greatly reduced by pulse compression as shown by Ref. [14]; thus, in this study direct we assume it is negligible.

3.2. Multistatic Detection Theory

Multiple authors describe the geometry of MSNs. Reference [11] analyzes the relationship between monostatic and bistatic active sonars. The author derives that a detection probability contour, that is, a contour consisting of all locations for a target t with the same detection probability, is defined by the constant product:

$$d_{t,s} \times d_{t,r} = \rho_{t,s,r}^2 \tag{1}$$

where $\rho_{t,s,r}$ is a constant known as the *equivalent monostatic range*, and $d_{t,s}$ and $d_{t,r}$ are the distances from a target t to a source s and to a receiver r, respectively (see Fig. 1). Those contours are geometric figures known as Cassini ovals, shown in Fig. 2. In Fig. 2, ρ_0 is the *range of the day*, defined as the distance from a monostatic sensor to a target where the detection probability is 50%.

There are multiple ways to model $P_{t,s,r}$, the probability of detecting target $t \in T$ with source $s \in S$ and receiver $r \in R$, as displayed in Fig. 3. The simplest is the definite range sensor model, which defines $P_{t,s,r}$ as

$$P_{t,s,r} = \begin{cases} 1 & \text{if } 0 \le \rho_{t,s,r} \le \rho_0 \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Then, the total detection probability for target t, P_t, considering all pairs of sources and receivers, is

$$P_t = \max_{(s,r)\in S\times R} P_{t,s,r}. \tag{3}$$

This means that if at least one pair of sensors detects target t, then $P_t = 1$, and otherwise $P_t = 0$. Although Eqs. (2) and (3) form an analytically convenient model, this model lacks some features of real sensors such as a gradually decreasing $P_{t,s,r}$ with increasing $\rho_{t,s,r}$. Hence, the DSTO team proposes two diffuse sensor models in Ref. [14]. The first is the Fermi function, defined here for a model that neglects the direct blast zone:

$$P_{t,s,r} = \begin{cases} \frac{1}{1+10^{(\rho_{t,s,r}/\rho_0-1)/b}} & \text{if } \rho_{t,s,r} \ge 0 \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

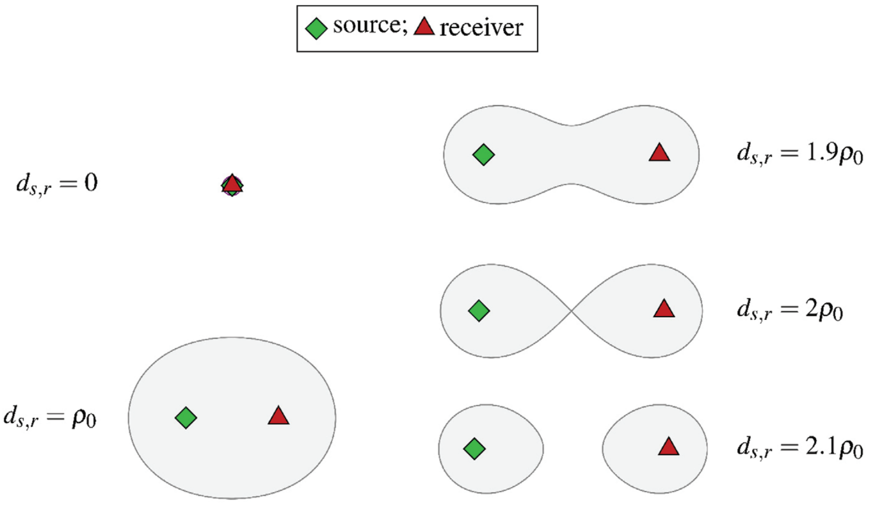

Figure 2. Examples of Cassini ovals—Cassini ovals are shown with respect to source and receiver distance $d_{s,r}$ where ρ_0 denotes the range of the day. If source and receiver are collocated, then the Cassini oval is a circle that corresponds to a monostatic sensor. Increasing the distance changes the shape from oval to two separate egg shapes. [Color figure can be viewed at wileyonlinelibrary.com.]

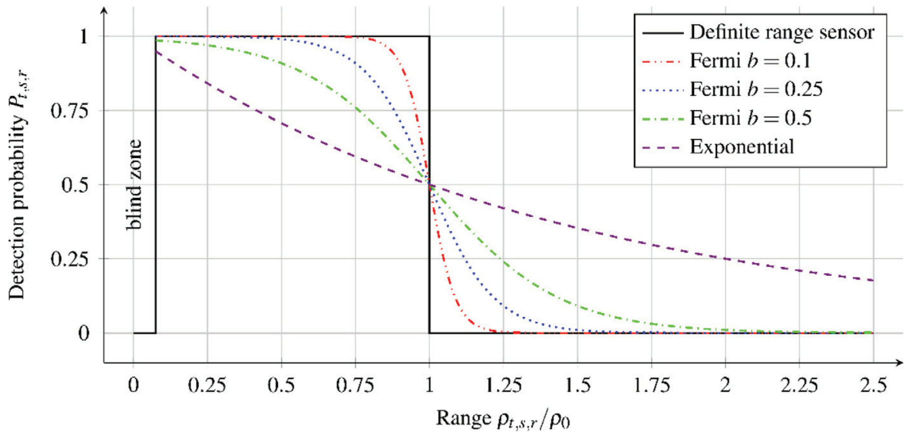

Figure 3. Sensor models—the probability curves for three sensor models: definite range, Fermi, and exponential. Range is expressed as multiples of the range of the day ρ_0. For $b > 0$, all models have $P_{t,s,r} = 0.5$ at $\rho_{t,s,r}/\rho_0 = 1$ to be consistent with the definition of the range of the day. In this figure, the "blind zone" denotes the range at which the direct blast effect occurs.

The diffusivity parameter b determines how rapidly probability values change with $\rho_{t,s,r}$. As $b \to 0$, the Fermi function approaches the definite range model.

The second model is the exponential function

$$P_{t,s,r} = \begin{cases} 10^{-0.30103\rho_{t,s,r}/\rho_0} & \text{if } \rho_{t,s,r} \geq 0 \\ 0 & \text{otherwise.} \end{cases} \quad (5)$$

Here, the exponent is chosen such that if $\rho_{t,s,r}/\rho_0 = 1$ the detection probability $P_{t,s,r} = 0.5$.

For diffuse sensor models such as these, we define a more general version of Eq. (4) to compute P_t. Assuming all probabilities are independent, we have

$$P_t = 1 - \prod_{(s,r)\in S x R} (1 - P_{t,s,r}). \quad (6)$$

In order to account for the relative importance of different targets, we introduce v_t as the value of target t. Hence, we denote the expected reward for target t as $v_t P_t$. If all targets have the same value, we simplify the expected reward for target t by using P_t. Based on this, we can define multiple objective functions depending on the chosen sensor model. While maximizing the total expected reward is our only objective when using a definite range model, we can

Table 1. Objective functions—different objective functions are possible depending on the user's goal and chosen sensor model, where v_t denotes the value or weight of target t

Objective	Sensor model	Formula
Maximize total (or average) expected reward	Definite range or diffuse	$\max \sum_{t\in T} v_t P_t$ or $\max \sum_{t\in T} v_t P_t/\lvert T\rvert$
Maximize minimum expected reward	Diffuse	$\max \min_{t\in T} v_t P_t$

also choose to maximize the minimum expected reward. The objective functions are summarized in Table 1.

4. OBSERVATIONS ON POINT COVERAGE SENSING

This section analyzes the point coverage scenario and makes some fundamental observations about the performance of a multistatic sonar system. These observations help to limit the solution space and exclude some infeasible and suboptimal solutions from consideration. We also show how to apply the gained insights to judge whether a particular outcome justifies utilizing a proposed method. Specifically, we consider the following questions:

- Is the convex hull spanned by the targets guaranteed to contain the optimal locations of sources and receivers?
- Are there other ways to exclude suboptimal solutions?

4.1. The Convex Hull

A simple approach to limit the set of potential sensor locations would be defining a rectangle that contains all targets. Even though this can easily be implemented, we want to exclude as many locations as possible. A more sophisticated approach involves looking at the convex hull spanned by the targets. As vividly described in Ref. [3], one can imagine the targets as nails sticking out of the plane. If we hold a rubber band around the nails and release it so that it stretches taut against the nails, the enclosed area will be the convex hull.

4.1.1. Properties of the Convex Hull

The example in Fig. 4 shows a convex hull for a set of targets T. It also shows that a target could reside at the vertex, along an edge or inside the convex hull of the resulting polygon. We define the set C as $C=\{t \in T: t$ is a vertex of the convex hull of the $T\}$.

The convex hull Conv(T) is defined as the smallest convex set that contains all targets in T. It is easy to see that Conv(T) = Conv(C). If all nails that are not vertices are removed in the previously mentioned example, the rubber band would still enclose the same area. Alongside this picturesque description of a convex hull, there also exists a mathematical definition. Each point $p \in$ Conv(C) with coordinates (x_p, y_p) can be written as a convex combination of the vertices in C. This is shown in Eq. (7), where $0 \le \lambda_t \le 1$ for all $t \in C$.

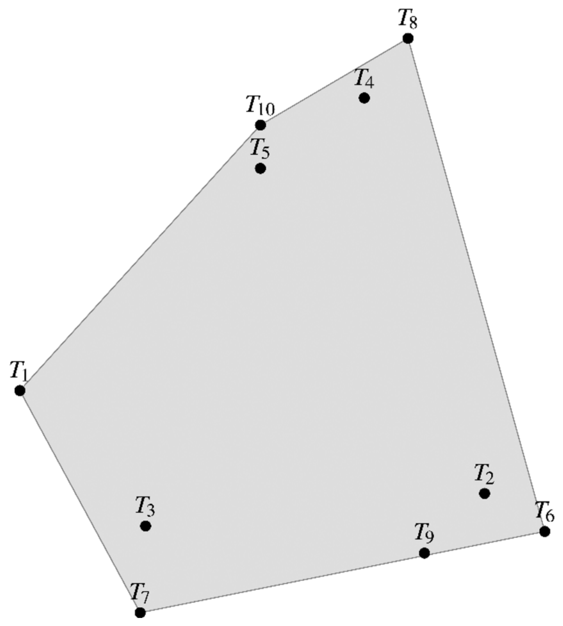

Figure 4. Convex hull example—a convex hull covering a set of targets. Its vertices are T_7, T_6, T_8, T_{10}, and T_1.

$$\begin{aligned} x_p &= \sum_{t\in C} \lambda_t x_t, \\ y_p &= \sum_{t\in C} \lambda_t y_t, \\ 1 &= \sum_{t\in C} \lambda_t. \end{aligned} \tag{7}$$

Theorem 1 describes the relationship between the convex hull and the detection probability for sensor p, where $P_t(p)$ denotes $P_{t,p,r}$ if p is a source and $P_{t,s,p}$ otherwise. For simplicity, we use the symbols p and p' to refer to both sensor and its position.

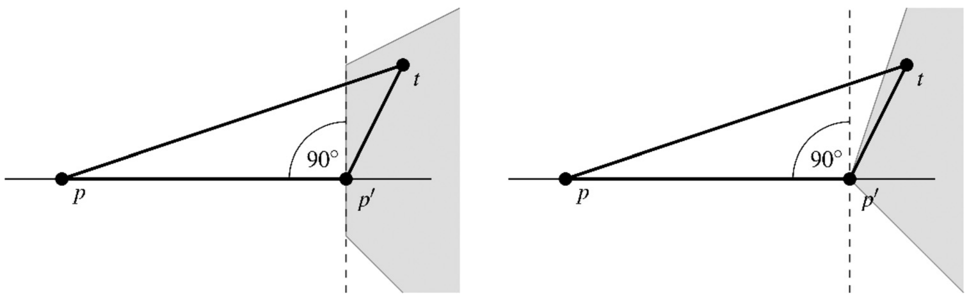

Figure 5. Theorem 1 proof—there exist two cases for the closest position $p \notin \text{Conv}(C)$ to p. The left figure displays p' on an edge of the convex hull, while p' is collocated with a vertex on the right. In both cases, all targets $t \in T$ are right of the dashed line, which is perpendicular to the line from p to p' and crosses p'.

THEOREM 1: For every potential sensor position $p \notin \text{Conv}(C)$, there exists a position $p' \in \text{Conv}(C)$ such that $P_t(p') \geq P_t(p) \forall t \in T$

PROOF: Assume sensor position $p \notin \text{Conv}(C)$. Let $p' \in \text{Conv}(C)$ denote the position inside the convex hull with the shortest distance to p. Then p' is either on an edge of the convex hull, or collocated with one of its vertices as shown in Fig. 5. In both cases, the convex hull and with it all targets $t \in T$ lie beyond an imaginary line perpendicular to the line from p to p' crossing that line at p'. Otherwise, there would be another p' that is closer to p, which is a contradiction.

Let t be a target inside the convex hull. Without loss of generality, we assume that p and p' are lying on a line parallel to the x axis. Then the horizontal distance between p and t is $|x_p - x_t| = |x_p - x_{p'}| + |x_{p'} - x_t|$, and we derive

$$\begin{aligned} d_{t,p}^2 &= (x_p - x_t)^2 + (y_p - y_t)^2 \\ &= (|x_p - x_{p'}| + |x_{p'} - x_t|)^2 \\ &\quad + (|y_p - y_{p'}| + |y_{p'} - y_t|)^2 \\ &\geq (x_{p'} - x_t)^2 + (y_{p'} - y_t)^2 = d_{t,p'}^2 \\ \therefore \quad d_{t,p}^2 &\geq d_{t,p'}^2. \end{aligned}$$

Let $\rho_t\ (p)$ denote $\rho_{t,p,r}$ or $\rho_{t,s,p}$ if p is a source or receiver, respectively. Thus, the equivalent monostatic range for p' is $\rho_t(p') = \sqrt{d_{t,p'}d_{t,x}} \leq \sqrt{d_{t,p}d_{t,x}} = \rho_t(p)$, where x is the second type of sensor needed in an MSN. It follows that $P_t(p') \geq P_t(p)$ for all sensor models that are monotonically nonincreasing with distance, as shown in Fig. 3. □

Theorem 1 implies that we can place all sources and receivers inside the convex hull of the targets without sacrificing optimality. We now describe a method to construct the convex hull.

4.1.2. Finding Vertices of the Convex Hull

Finding the vertices of a convex hull is a fundamental problem in computational geometry. There are several algorithms in literature; some of the most popular include: Graham scan [19], gift wrapping (sometimes called Jarvis march) [22], Andrew's monotone chain [1], quickhull [2], Chan's algorithm [8], divide and conquer [36], incremental convex hull [24], and the ultimate planar convex hull [29].

In this study, we adopt the Graham scan algorithm described in Ref. [19]. The algorithm runs in time $O(|T| \log |T|)$. It starts by finding $t_0 \in T$, the target with the smallest value for y_t. If there are multiple targets that share the smallest y_t, we pick the one with the smallest x_t out of the candidates. Next, we compute the angles θ_t each target $t \in T$ makes with t_0 and the x axis using the formula

$$\theta_t = \text{atan}^2(y_t - y_{t_0}, x_t - x_{t_0}) \tag{8}$$

Sorting the targets by θ_t from smallest to largest, the algorithm now considers each target as possible vertex. At each step, it determines whether the current target t and its two predecessors $t-1$ and $t-2$ make a turn clockwise or counterclockwise by calculating

$$\begin{aligned} c = (x_{t-1} - x_{t-2}) \times (y_t - y_{t-2}) - (y_{t-1} - y_{t-2}) \\ \times (x_t - x_{t-2}). \end{aligned} \tag{9}$$

If the result is positive, the points make a counterclockwise turn and the next target in the ordered list is considered. If $c<0$, then the turn is clockwise and the middle target $t-1$ is identified as non-vertex. If $c=0$, then all three targets are collinear; target $t-1$ is also discarded in this case. Figure 6 displays the working of the Graham scan algorithm.

The low complexity of the Graham scan is a result of the simple formula in Eq. (9) to detect clockwise and counterclockwise turns. Note that the Graham scan also outputs the vertices of the convex hull in a counterclockwise order; we exploit this fact in Section 5.2.4.

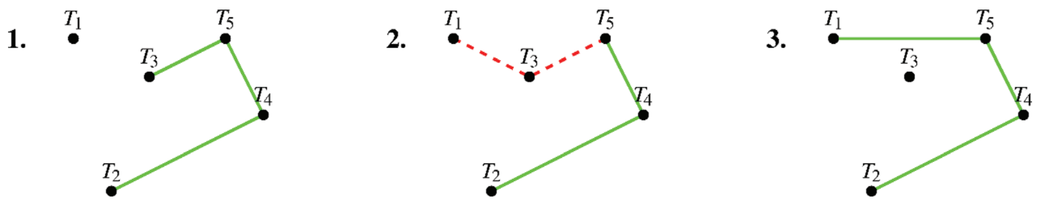

Figure 6. Graham scan—targets are considered in order of the angle they make between the lowest indexed target and the x axis. In this example, that order is T_2, T_4, T_5, T_3, then T_1. Step 1 shows the algorithm in progress, with only counterclockwise turns detected thus far. The algorithm detects a clockwise turn in step 2 and thus discards the middle target (T_3) in step 3. [Color figure can be viewed at wileyonlinelibrary.com.]

4.2. Range of the Day Circles

This section analyzes the relationship between the range of the day, ρ_0, and the definite range sensor model. We have seen Eq. (2) that a target is detected if and only if its equivalent monostatic range $\rho_{t,s,r} \leq \rho_0$. Building upon this fact, we introduce range of the day circles (RDCs). Each RDC is centered at a target and has radius ρ_0 as shown in Fig. 7. In the remainder of this section, we discuss methods to use RDCs to bound certain aspects of the sensor placement problem. Even though we develop these methods for definite range sensor models, they can also be extended to problems where we have to meet a particular detection probability goal.

4.2.1. Properties

An important observation is the relationship between sensor positions relative to RDCs and a target's detection probability.

LEMMA 1: A target t cannot be detected if no sensors are within t's RDC.

PROOF: To arrive at a contradiction, we assume all sensors are located outside the RDC of target t and $P_{t,s,r} = 1$ for some $s \in S$ and $r \in R$. Then, $d_{t,s} > \rho_0, \forall s \in S$ and $d_{t,r} > \rho_0, \forall r \in R$. It follows that

$$\begin{aligned}\rho_{t,s,r} &= \sqrt{d_{t,s} \times d_{t,r}}\\ &> \sqrt{\rho_0 \times \rho_0} = \rho_0\end{aligned}$$

But then according to Eq. (2), $P_{t,s,r} = 0$, which is a contradiction. □

This, however, does not mean that we can find an optimal sensor placement by solely looking at positions inside RDCs. The left plot in Fig. 8 demonstrates a counterexample where the optimal position for the source is located outside any RDCs. Moving this source inside an RDC results in losing coverage of targets.

A reasonable heuristic may involve choosing to place receivers within as many RDCs as possible and letting the sources be placed outside RDCs, thus "connecting" disparate elements of the MSN. However, the intuition that optimal receiver positions can only be found within maximal RDC intersections is fallacious, as illustrated by the right plot in Fig. 8. Receiver R_1 cannot be moved into the intersection of T_1's and T_2's RDCs without sacrificing coverage of T_1. Doing so would require moving the source closer to T_1, but this results in a loss of coverage of targets T_3 and T_4.

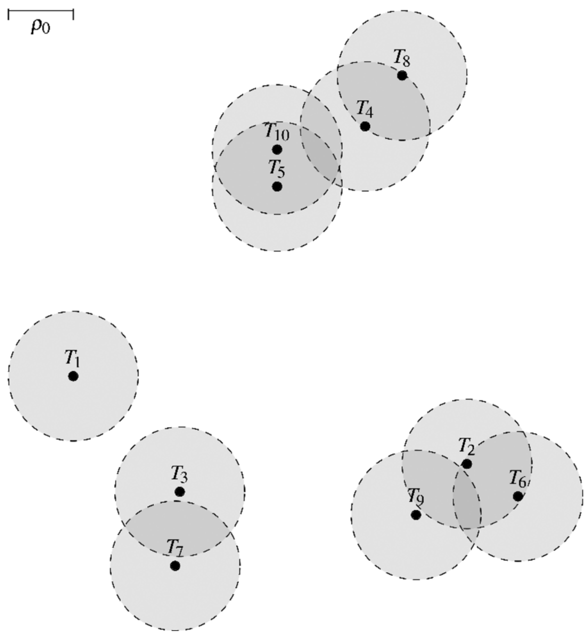

Figure 7. Range of the day circles (RDCs)—assuming a definite range sensor model, RDCs determine bounds for the problem. Each circle is centered at a target and has radius ρ_0.

4.2.2. Clusters

We have shown that we cannot restrict our sensors to occupy RDCs without potentially sacrificing optimality. Nevertheless, we can use RDCs and their intersections to assist our analysis. To do this, we introduce the notion of clusters. A cluster $G \subseteq T$ is a maximal set of targets, where the RDCs of all targets $t \in G$ mutually cover at least one point. Since

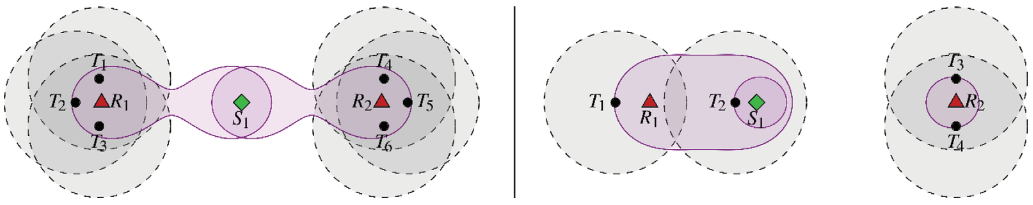

Figure 8. RDCs properties—the left plot illustrates that an optimal placement can be found by placing the source S_1 outside RDCs. Forcing the S_1 inside the RDCs results in losing coverage of one or more targets. The right plot shows an example where an optimal receiver position is outside RDC intersections. Here, it is not possible to cover all targets by placing all receivers into RDC intersections.

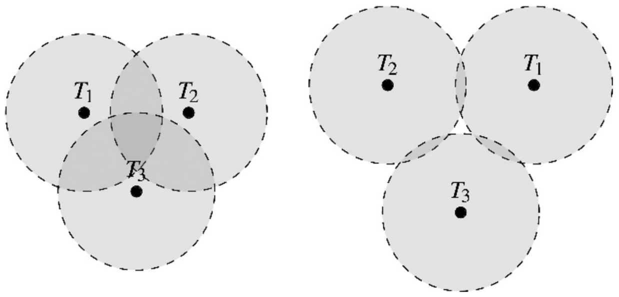

Figure 9. Clusters special case—the targets in the left plot form exactly one cluster. The targets on the right, however, form the three clusters $\{T_1, T_2\}, \{T_2, T_3\}$, and $\{T_1, T_3\}$ since the three targets do not have a single point in common.

we require a cluster to be maximal, it is not possible to add another target to a cluster. In our example from Fig. 7, we can find the clusters

$$G_1 = \{T_1\}, G_2 = \{T_3, T_7\}, G_3 = \{T_2, T_6, T_9\},$$
$$G_4 = \{T_4, T_5, T_{10}\}, G_5 = \{T_4, T_8\}.$$

We make some observations about clusters. First of all, each target $t \in T$ occurs in at least one cluster. If there are no intersections with circles from other targets, a target forms its own cluster, for example, T_1 is the only element of G_1. Additionally, it is possible that a target is a member of multiple clusters, for example, $T_4 \in G_4$ and $T_4 \in G_5$. Moreover, it is important to notice that even though at first glance it looks like the clique of targets on top of Fig. 7 (T_4, T_5, T_8, and T_{10}) all belong together, they actually form two separate clusters since T_8's RDC is only connected to T_4's RDC, not those of T_5 and T_{10}.

Figure 9 shows an interesting case: the left plot forms exactly one cluster, while the right plot forms three clusters $\{T_1, T_2\}, \{T_2, T_3\}$, and $\{T_1, T_3\}$ since the three targets do not have a single point in common, even though the targets' RDCs mutually intersect.

Based on this, we define a minimal set of clusters $\tilde{G}$ as the smallest set of clusters that contains all targets $t \in T$. For the right plot in Fig. 9, three minimal sets of clusters are $\tilde{G} = \{\{T_1, T_2\}, \{T_2, T_3\}\}$, $\tilde{G} = \{\{T_1, T_2\}, \{T_1, T_3\}\}$, and $\tilde{G} = \{\{T_1, T_3\}, \{T_2, T_3\}\}$. Furthermore, we formulate the following theorem.

THEOREM 2: A lower bound on the number of sensors required to detect all targets is $|\tilde{G}|$.

PROOF: To arrive at a contradiction, we assume that all targets are covered with $|\tilde{G}|-1$ sensors. It follows from Lemma 1 that at least one sensor has to be inside each target's RDC. Since the RDCs of each cluster have at least one point in common, putting a sensor on that point results in having a sensor in each RDC of this cluster. Moreover, since clusters are maximal, we cannot add another target to any cluster. Hence, one cluster and therefore at least one target remains without a sensor in its RDC. □

As a result of Theorem 2, we can define $|\tilde{G}|$ as a lower bound for the number of sensors required to detect all targets. By the same token, we are able to find an upper bound on the number of targets we can cover with a given number of sensors n by choosing n sets from $\tilde{G}$, such that the number of contained targets is maximized.

5. OPTIMIZING SENSOR PLACEMENTS

We now investigate methods to select sensor locations in MSNs. First, we formulate two mathematical programming models that select locations for a set of sources S and a set of receivers R. The first model, MSN-NLP, is an NLP that uses an arbitrary detection probability function $f(x_t, y_t, x_s, y_s, x_r, y_r)$. This function represents the probability of detecting target t with source s and receiver r, with the target located at coordinates (x_t, y_t), the source located at coordinates (x_s, y_s), and the receiver located at coordinates (x_r, y_r). Assuming each source and receiver pair detects each target independently of all other source and receiver pairs, MSN-NLP maximizes the expected value of the targets detected, where each target is associated with a weight reflecting either the importance of the location to be monitored, or the probability that a hostile enemy is present at the location. In the second model, MSN-INLP, we address the special case of a definite range sensor model using an INLP. Both MSN-NLP and MSN-INLP are nonconvex and thus may not prescribe a globally optimal solution.

In Section 5.2, we consider the problem of placing a single source in a network of fixed receivers. Although this problem is nonconvex, we are able to use the special structure of the problem to develop an algorithm that returns a provably near-optimal solution.

5.1. Mathematical Programming Formulations

Our mathematical models start from the premise that we have already found the set of vertices of the convex hull of the targets, C. They then select positions for a set of sources and receivers with the goal of maximizing the weighted probability of detection for the set of targets.

5.1.1. *Model MSN-NLP*

Indices and Sets:

$t \in T$	targets
$C \subseteq T$	vertex set, $C = \{t \in T \mid t \text{ is a vertex}\}$
$r \in R$	receivers
$s \in S$	sources
$p \in P = S \cup R$	all sensors (sources and receivers)

Data:

x_t	x coordinate of target t
y_t	y coordinate of target t
v_t	weight of target t

Decision Variables:

λ_t^p	weight of vertex t when defining sensor p's position
x_p	x coordinate of sensor p
y_p	y coordinate of sensor p

Objective function:

$$\max_{\lambda,x,y} \sum_{t\in T} v_t \left(1 - \prod_{s\in S,\, r\in R} (1 - f(x_t, y_t, x_s, y_s, x_r, y_r))\right) \tag{10}$$

Constraints:

$$x_p = \sum_{t\in C} \lambda_t^p x_t \quad \forall p \in P \tag{11}$$

$$y_p = \sum_{t\in C} \lambda_t^p y_t \quad \forall p \in P \tag{12}$$

$$1 = \sum_{t\in C} \lambda_t^p \quad \forall p \in P \tag{13}$$

$$\lambda_t^p \geq 0 \quad \forall t \in C, p \in P \tag{14}$$

MSN-NLP forces the positions of the sources and receivers to be inside the convex hull by defining them as convex combinations of the vertices in Eqs. (11)–(14). The objective function (10) calculates sum of the overall probabilities of detecting each target t, weighted by t's relative value.

5.1.2. *Model MSN-INLP*

For the case of a definite range sensor model, we have $f(x_t, y_t, x_s, y_s, x_r, y_r)$ as a step function, which may be problematic for many solvers. Thus, we propose model MSN-INLP, which uses binary decision variables to implement the sensor model. First, we introduce binary decision variables $h_{t,s,r}$. For each target t, $h_{t,s,r}$ specifies whether t is detected by source s and receiver r ($h_{t,s,r} = 1$) or not detected by source s and receiver r ($h_{t,s,r} = 0$). Equation (19) ensures that $h_{t,s,r}$ does not indicate a detection for any combination of target, source, and receiver that does not satisfy the requirement stated in the definite range sensor model. The target can only be denoted as detected ($h_{t,s,r} = 1$) if $\sqrt{d_{t,s} \times d_{t,r}} \leq \rho_0$. If the target is not detected, then a large number M_t is added to the range of the day, ρ_0, in order to satisfy the constraint. We then use binary decision variables k_t to specify whether each target t is detected by any source and receiver. Equation (20) ensures that if a target t is not detected by any source and receiver ($h_{t,s,r} = 0$ for all s and r), then $k_t = 0$. Otherwise, k_t is allowed to take on value 1. Equations (16)–(18) and (21) are identical to Eqs. (11)–(14); again, they ensure that the sources and receivers are placed within the convex hull of the targets.

Additional Data:

ρ_0	range of the day
M_t	constant sufficiently large to satisfy Eq. (19)

Additional Decision Variables:

$h_{t,s,r}$ binary; equal to 1 if target t is detected using source s and receiver r, 0 otherwise

k_t binary; equal to 1 if target t is detected using any source and receiver, 0 otherwise

Objective function:

$$\max_{\lambda,x,y,h,k} \sum_{t\in T} v_t k_t \tag{15}$$

Constraints:

$$x_p = \sum_{t\in C} \lambda_t^p x_t \quad \forall p \in P \tag{16}$$

$$y_p = \sum_{t\in C} \lambda_t^p y_t \quad \forall p \in P \tag{17}$$

$$1 = \sum_{t\in C} \lambda_t^p \quad \forall p \in P \tag{18}$$

$$\rho_0 + M_t(1 - h_{t,s,r}) \geq \left(\sqrt{(x_t - x_s)^2 + (y_t - y_s)^2} \times \sqrt{(x_t - x_r)^2 + (y_t - y_r)^2}\right)^{\frac{1}{2}} \quad \forall t \in T, s \in S, r \in R \tag{19}$$

$$k_t \leq \sum_{s \in S,\, r \in R} h_{t,s,r} \quad \forall t \in T \tag{20}$$

$$\lambda_t^p \geq 0 \quad \forall t \in C, p \in P \tag{21}$$

$$k_t,\ h_{t,s,r} \in \{0,1\} \quad \forall t \in T,\ s \in S,\ r \in R \tag{22}$$

Reference [6] shows that tight parameters such as M_t reduce the time a solver needs to find an optimal solution. In order to make sure the INLP is always feasible, the left-hand side of Eq. (19) has to be equal to the largest product of distances possible for each target. Since the largest distance from a target position to a point in a convex hull is to one of its vertices, we can set M_t to its square. We can also subtract ρ_0 since it is already added on the left-hand side. Thus, we have

$$M_t = \max_{t' \in C} d_{t,t'}^2 - \rho_0. \tag{23}$$

Note that although both MSN-NLP and MSN-INLP are relatively easy to specify, they are quite difficult to solve to optimality. Both are highly nonconvex, and even in the simple bistatic case where $|S| = 1$ and $|R| = 1$, there is always more than one optimal solution. First of all, interchanging the source and receiver in a bistatic environment does not change the outcome; the Cassini ovals are unchanged by reversing the sensor types. In the case of a definite range sensor, if a sensor is moved slightly in one direction, it may still detect the same targets. Indeed, there often exist infinitely many optimal solutions, as well as many suboptimal local maxima.

While the problem of placing multiple sources and receivers is quite difficult, the special case in which the receiver positions are already fixed and we only wish to place a single source is more tractable. Although this simpler problem is still nonconvex, we now develop an algorithm that is guaranteed for find a location for which the objective value is within a predetermined $\epsilon > 0$ of the global optimal objective value, where ϵ is chosen by the user.

5.2. DiBS Algorithm

In this section, we consider the optimal placement of a single source in a network of fixed receivers. In an operational setting, it often occurs that receiver sonobuoys have been deployed throughout a field by plane or ship, and source pings are to be conducted one at a time via helicopter dips. In this setting, it is useful to consider the best location for the next (and possibly only) ping.

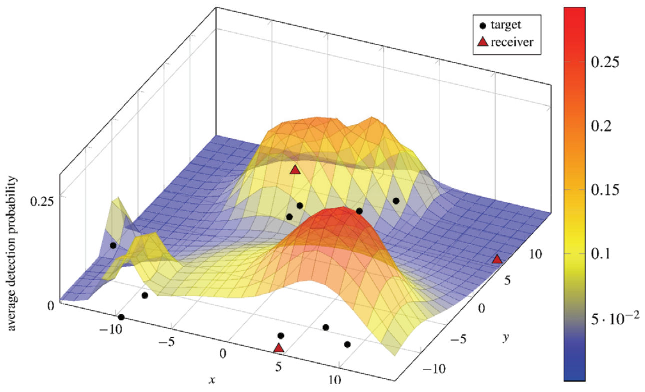

Figure 10. DiBS' problem statement—we wish to find the optimal source position, given fixed target, and receiver locations. Our goal is to maximize the average detection probability across all targets. The z axis represents the average detection probability assuming a source is deployed at the respective position. Multiple local maxima indicate nonconvexity.

5.2.1. *Problem Statement*

We consider the following scenario: there is a set T of fixed target locations that have to be monitored. Furthermore, there is a set R of receivers with fixed and known positions. The objective is to find the optimal position for a single source. We do not assume a particular sensor model; rather, we discuss a method that works with all sensor models of the form shown in Fig. 3, that is, those in which the detection probability is nonincreasing with the equivalent monostatic range.

Figure 10 shows a generic scenario with fixed targets and receivers. For each position (x, y), the vertical axis represents the average detection probability (averaged over all targets) if a single source is deployed at position (x, y). More generally, we can easily incorporate the target weights v_t to calculate an objective value for each location. Multiple local maxima in this plot show that the problem is nonconvex. Nonetheless, we now develop an algorithm guaranteed to solve this problem to near global optimality.

5.2.2. *Algorithm Development*

In this section, we develop the DiBS algorithm. DiBS partitions the area of possible source locations into sectors. For each sector, we calculate an upper bound on the objective value that could be obtained if a source were to be placed within that sector. This upper bound is generally loose, but it provides an (optimistic) estimate of the best objective value that can be obtained by placing a source within the sector. We can also determine a lower bound on the optimal objective value by considering any feasible solution, for example, by placing the source at the center of a sector. In each iteration, the sector with the highest upper bound is divided into multiple smaller sectors, and we calculate an upper bound for

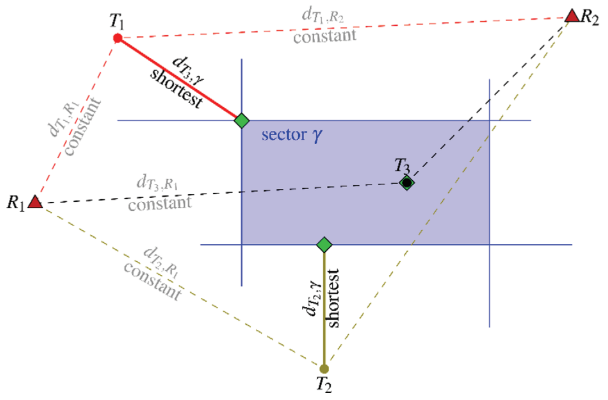

Figure 11. Determine upper bound—for each target t there exists a hypothetical source position inside sector γ that has the shortest distance $d_{t,\gamma}$ to the target. This distance determines the highest possible detection probability with a source position inside γ. Using the probabilities for all targets, we can calculate an upper bound for the objective value if the source is placed inside γ.

each of these smaller sectors. These upper bounds are generally tighter than the bound obtained from a larger sector. In this way, we perform more and more detailed examinations of the most promising sector. This process continues until we meet some termination criterion. For instance, we may wish to stop when we have a feasible solution with an objective value that is "close enough" to our current best upper bound on the optimal objective value.

The DiBS algorithm can be categorized as a divide-and-conquer algorithm. Divide-and-conquer type algorithms are commonly used in sorting [30], multiplying large numbers [27], Eigen decomposition of symmetric tridiagonal matrices [20], finding the closest pair of points [37], construction of Voronoi diagrams [28], and computing the discrete Fourier transform [16]. Such algorithms are generally developed to solve large or difficult problems. They work by recursively reducing the problem into smaller and smaller sub-problems, until the sub-problems are small enough to be solved (conquered) directly. These solutions are then combined into the solution for the original problem [10].

In Section 4.1, we showed that optimal sensor positions can always be found inside the convex hull spanned by the targets. As a practical matter, we slightly relax this condition to the smallest rectangle with edges parallel to the x and y axes that contains all targets. This way we can easily divide the area of possible source locations into smaller sectors by slicing horizontally as well as vertically.

Let Γ denote the set of sectors in which the source may be placed. In each iteration of DiBS, each sector $\gamma \in \Gamma$ is evaluated with respect to an upper bound, uB(γ), for the objective value. This is the heart of DiBS and is illustrated in Fig. 11. Since receivers and targets occupy fixed positions in this scenario, each target-receiver distance $d_{t,r}$ is constant. Thus, the equivalent ranges $\rho_{t,s,r}$ and subsequently the detection probability P_t only depend on the target-source distance $d_{t,s}$. Hence, for every target, we determine the hypothetical source position that is closest to the target but still inside γ. These hypothetical source positions are either on the edge of the sector or the target position itself, as seen in Fig. 11.

With the shortest distance $d_{t,\gamma}$ from target t to its respective hypothetical source position inside sector γ, we can compute the highest possible detection probability P_t, assuming the source location is inside γ. Having done this with all targets, we compute the upper bound, uB(γ), according to Table 1. The following pseudocode finds the upper bound for a given sector γ, where $x_\gamma^{\min}$ ($y_\gamma^{\min}$) is the lowest and $x_\gamma^{\max}$ ($y_\gamma^{\max}$) the highest x (y) coordinate for γ.

```
1: procedure UPPER.BOUND($\gamma$)
2:     for all $t \in T$ do
3:         $x_s \leftarrow \min(\max(x_\gamma^{\min}, x_t), x_\gamma^{\max})$
4:         $y_s \leftarrow \min(\max(y_\gamma^{\min}, y_t), y_\gamma^{\max})$
5:         $d_{t,\gamma} \leftarrow \sqrt{(x_t - x_s)^2 + (y_t - y_s)^2}$
6:         compute $P_t$                    ▷ use chosen sensor model
7:     end for
8:     compute objective value $Z$     ▷ use chosen objective
9:     return $Z$
10: end procedure
```

In a next step, we pick the sector $\gamma \in \Gamma$ in which uB(γ) is maximized and divide it into smaller sectors with respective upper bounds. Consistently repeating this method results in smaller sectors and subsequently tighter upper bounds until a termination condition is met. Various termination conditions are conceivable, such as a maximum sector size, optimality range, etc. The following pseudocode shows the workings of DiBS.

```
1: procedure DiBS
2:     create initial $\Gamma$
3:     $optimal \leftarrow$ FALSE
4:     while $\neg$ $optimal$ do
5:         select $\gamma \in \Gamma$ such that UPPER.BOUND($\gamma$) is
           maximized
6:         if termination condition is met then
                                    ▷ use chosen termination condition
7:             $optimal \leftarrow$ TRUE
8:         else
9:             $\Gamma' \leftarrow$ split $\gamma$ into smaller sectors
10:            $\Gamma \leftarrow \Gamma' \cup \Gamma \setminus \{\gamma\}$
11:        end if
12:    end while
13:    return $\gamma$
14: end procedure
```

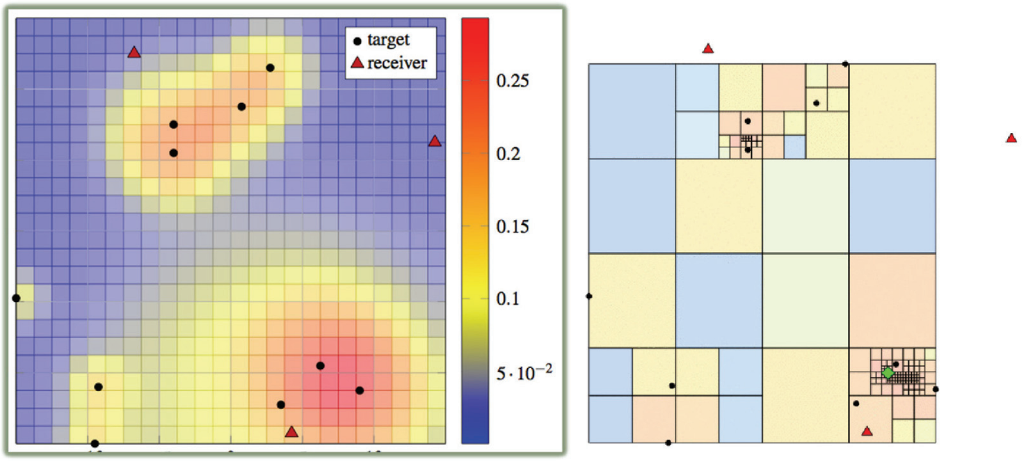

Figure 12. DiBS example—a scenario with 10 targets and three receivers as given in Fig. 10. On the left, we see the objective value as a function of source position. Note that there are two main areas of good objective value: the upper center and the lower right. A large number of targets coincides with a receiver in each of these locations. On the right, we see the final state of DiBS. Note that DiBS has performed a thorough search of the two promising areas, while leaving less promising areas relatively unexplored. The green diamond denotes DiBS' final solution.

Figure 12 shows the end state of an instance where example from Fig. 10 is solved with DiBS. It vividly illustrates how sectors with a low upper bound are ignored while promising areas are further explored. In the remainder of this section, we examine various details of DiBS and give recommendations on how to use it most efficiently. We end with a brief computational study examining DiBS' computation time.

5.2.3. *Termination Conditions*

First, we wish to verify that the algorithm eventually finishes when it meets a specific condition. This section discusses various termination conditions based on sector size and optimality gap.

Termination by Sector Size. There are two natural ways to define the size of a rectangular sector: by its area and by its edge lengths. A small area, however, can be the result of a very narrow but long rectangle. This metric is not very useful, because detection probabilities can change greatly along a line, as well as in a long rectangle. Thus, even if a sector with small area has a high upper bound, a sensor placed in this sector may perform badly. Additionally, telling a decision maker to place the source in a 0.01 $\times$ 10,000 foot rectangle is likely not useful in practice.

On the other hand, using a maximum edge length strongly constrains possible source locations. Choosing a maximum edge length of ten feet, for example, results in a final sector that has size of at most 10 $\times$ 10 feet, a sufficient precision for helicopter dips. Moreover, the change in detection probabilities in such a region is very limited. Therefore, for the rest of this section, "Termination by Sector Size" denotes termination when the length of the longest edge of the sector with the highest upper bound is sufficiently small.

Table 2. Optimality gap methods—these methods look for a specific source position inside a sector in order to determine the optimality gap. The last method, *Targets*, has to be used in conjunction with one of the other methods since there may exist sectors containing no targets

Method	Positions	Remarks
Center	1	Uses sector's center point
Corners	4	Uses sector's vertices
Corners + Center	5	Combines Corners and Center
Targets	up to $\|T\|$	Uses target positions inside sector

Termination by Optimality Gap. Though simple and useful, it turns out that termination by sector size has a drawback. If we request a sector with very short edges, DiBS may expend considerable computation time in order to obtain a negligible gain in objective value. Hence, we now consider a termination condition with a technique used by many mathematical programming solvers: we introduce a tolerable optimality gap, ϵ. That is, we allow DiBS to stop as soon as it finds a feasible sensor position s' that has an objective value within ϵ percent of the upper bound. The effectiveness of the optimality gap, turn, depends on which position we choose for s'.

Table 2 contains a number of methods for finding a feasible source position from which to create a lower bound. The first method, *Center*, places the source at the center of the most promising sector. It does not require high computational power, but it generally produces a looser bound than

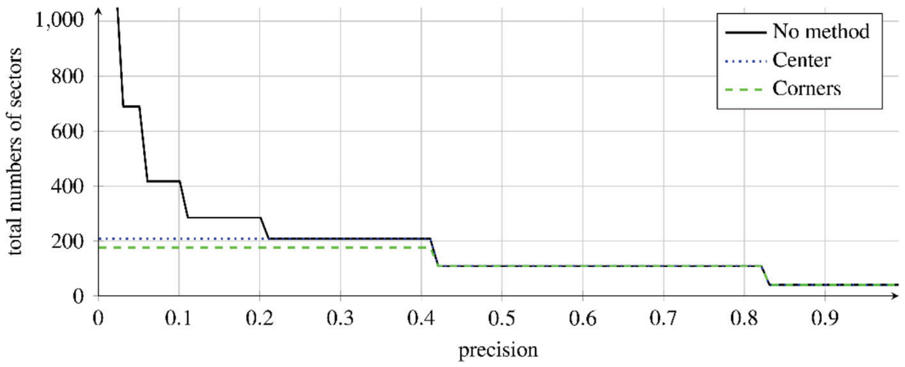

Figure 13. Optimality gap example—while the number of evaluated sectors for the baseline case without optimality gap feature increases with higher demanded precision, the algorithm terminates earlier when using an optimality gap. [Color figure can be viewed at wileyonlinelibrary.com.]

the other methods. If we consider five positions with *Corners + Center*, we generally obtain a tighter bound but with increased run time. The last method, *Targets*, is an add-on for the previous ones: in addition to any previously considered position it also considers placing the source so as to coincide with any targets in the sector. The idea is that the detection probability for a single target is maximized if at least one sensor is at the target's position. Since there are sectors that do not contain any targets, it can only be used in conjunction with another method.

The example in Fig. 13 shows how an optimality gap limits the number of created and evaluated sectors and thus the run time. In this particular problem instance, DiBS creates more than 16,000 sectors to terminate with a sector size of 0.001. Utilizing an optimality gap of $\epsilon = 0.05$, however, DiBS terminates after about 200 created sectors with only a minor difference between Center and Corners. Overall, we recommend using a termination condition based on an optimality gap rather than a sector size. In our experience, such a termination condition significantly reduces the number of created and evaluated sectors, especially for high precision calculations. Even though the run time per iteration slightly increases, the total run time can be reduced to a fraction of the run time required to achieve similar accuracy using a sector size criterion, regardless of the choice of method for finding the feasible source position s [31].

Utilizing Lower Bounds Recall that the methods described in the previous section provide a lower bound, $\text{lB}(\gamma)$. This section discusses whether we can use lower bounds to disregard sectors. Additionally, we examine the performance of DiBS using lower bounds in this manner.

Consider discarding a hypothetical sector γ' with upper bound $\text{uB}(\gamma')$ that is less than or equal to the lower bound $\text{lB}(\gamma)$ of another sector γ. Interestingly, this is not necessary. Since $\text{lB}(\gamma)$ is the objective value for a feasible source position in sector γ, it is always included in one sector of γ's subsectors following a division. Hence, there is always a subsector of γ that has a higher upper bound than γ'. Thus, γ' will never be chosen as the best sector, and therefore, discarding it does not reduce the total number of sectors created. It does, however, reduce the number of sectors in a list that has to be searched for the highest upper bound. However, the cost of this operation is negligible compared to other operations in DiBS.

Nevertheless, we can use the lower bound to indirectly measure the tightness of the upper bound. A tight upper bound is important for faster identification of a solution. Figure 14 illustrates the evolution of the gap between the upper and lower bounds throughout the execution of DiBS. Here, the left plot shows the overall trend of a decreasing gap the longer the algorithm runs. Jumps to lower values for the lower bound occur when the next sector with highest upper bound is much larger than the previous sector and therefore can provide a bigger gap between lower and upper bound. (The lower bound at each iteration is that resulting from the current sector under consideration, not the greatest lower bound found so far.) The right plot verifies our suspicion that smaller sectors yield smaller gaps and therefore tighter upper bounds.

5.2.4. *Implementation Issues: Plane Rotation*

Various practical issues arise when implementing DiBS. Here, we examine the setup of the initial set of sectors. Reference [31] describes an approach for handling rectangular sectors that are not squares.

The DiBS algorithm starts by finding the smallest rectangle with edges parallel to the x and y axes that contains all targets. We know from Section 4.1 that the optimal sensor position lies inside the convex hull spanned by the targets. Hence, it is possible that the resulting rectangle contains a large overhang that is subject to being evaluated unnecessarily. This section discusses the use of the minimum-area rectangle containing

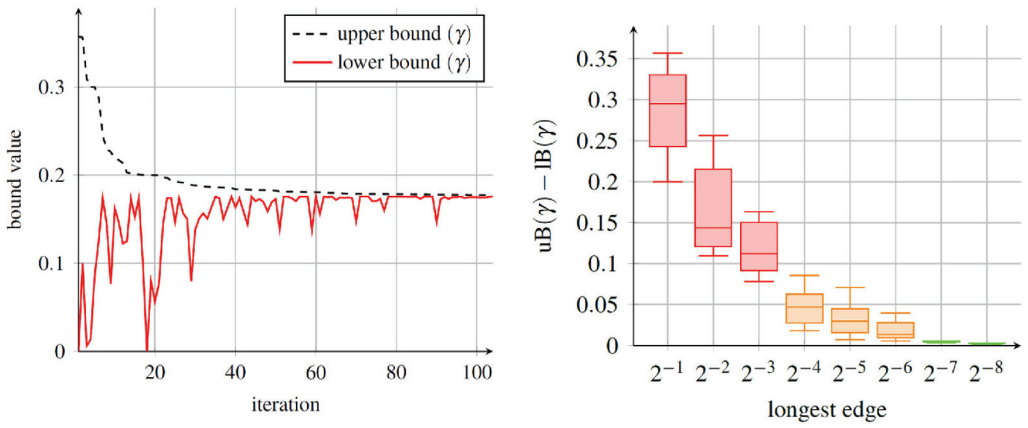

Figure 14. DiBS goodness—with increasing iterations the gap between the upper and lower bounds decreases (left plot). Steep drops in lower bound values occur in iterations where DiBS begins to evaluate a large sector. Overall, however, the bounds become closer together as the iterations proceed, reflecting the increasing fidelity of DiBS' approximation of the objective function. The right plot shows the relationship between longest sector edge length (expressed as a multiple of the longest initial edge length) and gap size. Again, smaller sectors yield tighter upper bounds. [Color figure can be viewed at wileyonlinelibrary.com.]

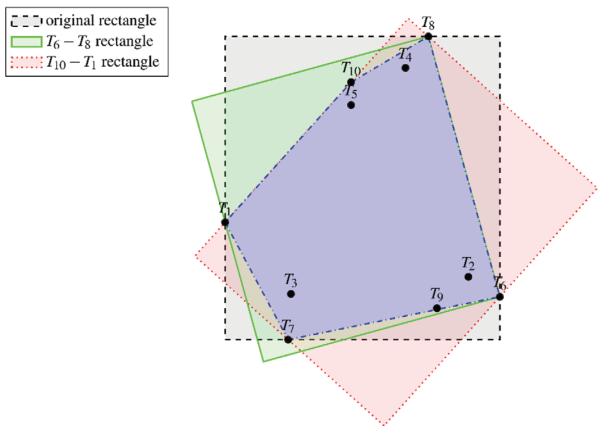

Figure 15. Minimum-area rectangle example—the plot shows the original rectangle as well as two rectangles that have one side collinear with one side of the convex hull. The T_6– T_8 rectangle is the minimum-area rectangle that encases all targets. [Color figure can be viewed at wileyonlinelibrary.com.]

all targets and how to rotate the plane such that our original implementation of DiBS still can be applied.

Reference [15] proves that the minimum-area rectangle encasing a convex hull has a side collinear with one of the edges of the convex hull, as seen in Fig. 15. Hence, we choose the rectangle with smallest area out of the $|C|$ possible rectangles we could form. Using the Graham scan to find C has the advantage that the vertices are already ordered. Therefore, we can use this output to easily construct the edges of the convex hull.

The example in Fig. 15 shows the original rectangle produced by DiBS as well as the rectangles with the smallest and the largest area that have one side collinear with one of the edges of the convex hull. The respective areas in square units are

$$\text{original} : 636, \quad T_6 - T_8 : 505.375, \quad T_{10} - T_1 : 619.420.$$

This example shows that using the minimum-area rectangle can greatly reduce the overhang. Having determined the minimum-area rectangle, we need to rotate the plane such that the edges of the rectangle are parallel to x and y axes. An efficient way to do this is applying a linear transformation to the sets of target and receiver locations [32]. Let t_0– t_1 be the edge that is collinear with the minimum-area rectangle. Then, the linear transformation is represented by the matrix

$$A = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}, \text{where } \theta = \tan^{-1}\left(\frac{y_{t_1} - y_{t_0}}{x_{t_1} - x_{t_0}}\right).$$

The matrix multiplication AX provides the new coordinates for targets and receivers, where the first and second row of X holds the x and y coordinates, respectively.

Although plane rotation can have a huge impact on the size of the overhang, this may not necessarily result in a corresponding increase in DiBS iterations, since the overhang may be largely devoid of targets. Nevertheless, since the preprocessing required to choose a small rectangle is not very time-consuming, one should consider executing this step before running DiBS.

5.2.5. *Finding Multiple Sensor Locations*

In some applications, it is desirable to optimize the positions of more than one source, for example, when placing multiple sonobuoys. This section investigates an iterative method as well as a simultaneous approach for accomplishing this.

Placing Sources Iteratively. We consider a set S of sources that has to be optimally deployed in a scenario with fixed targets and receivers. Furthermore, we assume a definite range sensor model in this section, although other models are possible. We choose the position for the first source by running DiBS without modifications. In a next step, we remove all targets that are detected in the first iteration and run DiBS again with the remaining targets to choose the second sensor position. We repeat those until either all targets are detected or all sources are placed. The following pseudocode illustrates the iterative method.

```
1: procedure ITERATIVE.METHOD
2:     while $|T|>0$ and $|S|>0$ do
3:         select $s \in S$
4:         run DiBS
5:         $T \leftarrow T \setminus \{\text{detected targets}\}$
6:         $S \leftarrow S \setminus \{s\}$
7:     end while
8: end procedure
```

Since we place sources in a greedy manner, the resulting solution generally is not optimal. Nevertheless, this algorithm reveals some information about the optimal solution. First of all, if all targets are detected by this method, the solution indeed is optimal. Otherwise, we can use the outcome to define lower and upper bounds on the optimal objective value for this problem by noting that the objective value achieved by DiBS is a *submodular* function [33] of the set of source locations selected. This property is easiest to see in the context of a definite range sensor model, so we continue our discussion in that setting. However, a similar argument applies in the case of a continuous sensor.

If Ω is a set, a function g: $2^{\Omega} \rightarrow \mathbb{R}$ is submodular if and only if

$$g(X \cup \{j,k\}) - g(X \cup \{k\}) \leq g(X \cup \{j\}) - g(X)$$
$$\forall\, X \subseteq \Omega \text{ and } j,k \in \Omega \setminus X.$$

In the multistatic sonar setting, let Ω be the (infinite) set of possible locations for sources, and let $D(X)$ denote the set of all targets detected as a function of the set X of source locations selected. Recall that the set of targets detected by a source placed at location j depends only on the locations of the receivers and not on the presence or location of any other sources. Thus, given an initial set of locations X, adding a new source at location j might add several new targets to the detectable set. If sensor location k is added before j, some of the targets added by j might have already been added by k. Therefore, we have $D(X \cup \{j,k\}) \setminus D(X \cup \{k\}) \subseteq D(X \cup \{j\}) \setminus D(X)$. Because $g(X) = |D(X)|$, this implies that g is a submodular function. As shown in Refs. [12] and [33], an iterative greedy algorithm has an optimality gap of at most $1 - e^{-1} \approx 0.632$ when maximizing a submodular function. Hence, we define the lower bound, lB, as the objective value achieved by the iterative approach and the upper bound as $\text{uB} = \frac{\text{lB}}{1-e^{-1}} \approx 1.582 \times \text{lB}$. We conclude that even though the iterative method does not guarantee an optimal solution, it is a quick way to get an estimate of the quality of other solutions. Combined with our observations from Section 4.2, we are able to narrow down the optimal solution even more.

Placing Sources Simultaneously. In contrast to the iterative method, one might also consider placing multiple sources simultaneously. One simple method would involve placing each source in an arbitrary sector and calculating an upper bound on the objective value by assuming that *all* sources are as close as possible to each target, within the bounds of their respective sectors. In order to exhaust all possible solutions, one could evaluate all combinations of sectors, taking into account the fact that multiple sources might occupy the same sector. In this case, the number of possible combinations is calculated by combinations of multisets [5] as

$$\binom{|S| + |\Gamma| - 1}{S|} = \frac{(|S| + |\Gamma| - 1)!}{|S|!(|\Gamma| - 1)!} \tag{24}$$

where $|S|$ is the number of sources and $|\Gamma|$ the number of sectors in a particular iteration. The base case with one source represents the original DiBS where the number of combinations equals the number of sectors, that is, we consider placing the source in each sector. With two sources and four sectors, we have the following 10 combinations.

$1: \{\gamma_1, \gamma_1\}$	$2: \{\gamma_1, \gamma_2\}$	$3: \{\gamma_1, \gamma_3\}$	$4: \{\gamma_1, \gamma_4\}$
$5: \{\gamma_2, \gamma_2\}$	$6: \{\gamma_2, \gamma_3\}$	$7: \{\gamma_2, \gamma_4\}$	$8: \{\gamma_3, \gamma_3\}$
$9: \{\gamma_3, \gamma_4\}$	$10: \{\gamma_4, \gamma_4\}$		

Clearly, the number of combinations grows very rapidly with the number of sectors and sources. While placing just a few sources simultaneously in this manner might be manageable, using this approach with many sensors quickly becomes impossible. This is aggravated by the fact that selecting and dividing multiple sectors per iteration increases the number of sectors and subsequently the number of combinations even more quickly. A future study might find ways to reduce that number or find another method to deal with multiple sources.

5.2.6. *Computation Time of DiBS*

We now conduct computational experiments to assess the performance of DiBS in terms of computing time. We use R (x64) version 3.3.1 to generate 100 problem instances

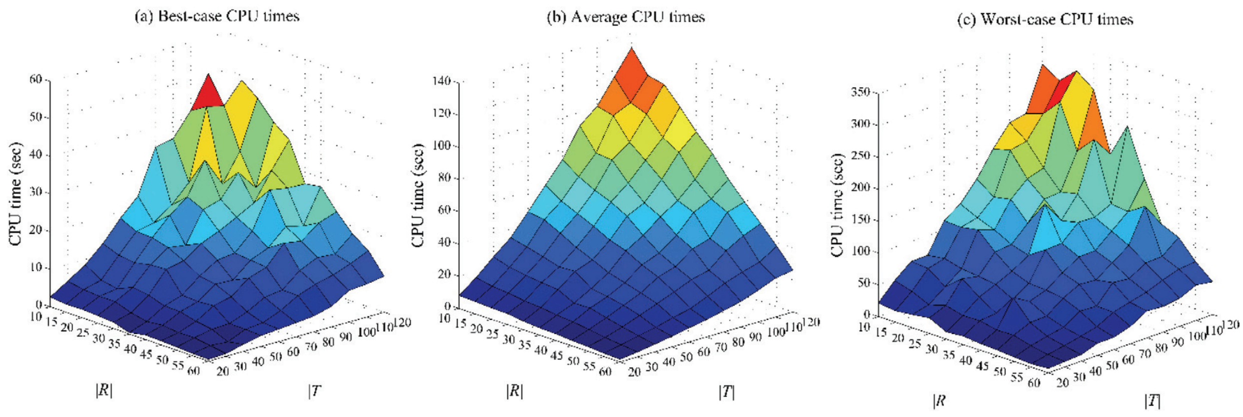

Figure 16. (a) Best-case CPU times, (b) average CPU times, (c) worst-case CPU times—the average computation time for DiBS increases with both the number of receivers and the number of targets, but remains less than three minutes for all problem sizes. Note that each plot uses a different vertical axis scale.

for each problem size we consider. We consider problem instances with varying number of receivers and targets, ranging from $(|R|, |T|) = (10, 20)$ to $(|R|, |T|) = (60, 120)$. For each instance, we generate receiver and target locations uniformly at random in a 10×10 unit two-dimensional area. Next, we solve the source placement problem with the DiBS algorithm using the "center" gap method with a 5% optimality gap, and we choose $\rho_0 = 1$ and $b = 0.25$. All experiments are run on a computer with two Intel® Xeon® CPU E5-2687W 3.10 GHz processors and 128 GB RAM.

The computation time results are shown graphically in Fig. 16. Figure 16a–16c display the best-case, average, and worst-case CPU times (in seconds) of DiBS for all problem sizes. The figure indicates that the computation time of DiBS increases with the number of receivers and targets. For example, DiBS can solve a small problem with $(|R|, |T|) = (30, 30)$ in less than 10 seconds (with an average of 8 seconds), a medium-sized problem with $(|R|, |T|) = (40, 90)$ in less than 125 seconds (with an average of 57 seconds), and a relatively large problem with $(|R|, |T|) = (60, 120)$ in less than 330 seconds (with an average of 131 seconds). All of these computation times are acceptable in many practical applications. Moreover, both number of receivers and number of targets have similar impact on the computing time of DiBS.

6. CONCLUSIONS

This article considers the problem of optimally placing multistatic sonar sensors. Although such sensors enjoy extensive use in practical applications, they appear only infrequently in the published literature. We first develop some novel insights that help bound the problem. In particular, we show that placing sensors inside the convex hull that encases all targets results in detection probabilities as least as good as any placements outside the convex hull. This important observation limits the area where we have to search for optimal sensor positions regardless of the sensor model we use.

Assuming a definite range sensor model, we show that we can utilize clusters of RDCs to bound our equipment requirements. The number of elements in the minimal set of clusters, $\tilde{G}$, represents a lower bound on the number of sensors required to cover all targets. In addition, with a fixed number of sensors, n, we are able to determine an upper bound on the number of targets covered by selecting n clusters from $\tilde{G}$, such that the number of covered targets is maximized.

In a subsequent step, we formulate two mathematical programming models to find optimal positions for a set of sources and receivers. These nonconvex formulations do not yield provably good solutions, even if the positions of one type of sensor are fixed. Hence, we develop the new DiBS algorithm. Assuming fixed receiver positions, this algorithm is the first in the literature to provide a near-optimal solution to the nonconvex single-source placement problem, and our computational results indicate that it is also quite efficient for solving realistically-sized problems.

ACKNOWLEDGMENTS

The authors thank Prof. Craig Rasmussen of the Naval Postgraduate School for many helpful discussions. Prof. Craparo is funded by the Office of Naval Research.

REFERENCES

[1] A.M. Andrew, Another efficient algorithm for convex hulls in two dimensions, Inf Process Lett 9 (1979), 216–219.
[2] C.B. Barber, D.P. Dobkin, and H. Huhdanpaa, The Quickhull algorithm for convex hulls, ACM Trans Math Softw 22 (1996), 469–483.

[3] M.D. Berg, M. Van Kreveld, M. Overmars, and O.C. Schwarzkopf, Computational geometry, Springer, Berlin Heidelberg, 2000.
[4] J.I. Bowen and R.W. Mitnick, A multistatic performance prediction methodology, Johns Hopkins APL Tech Dig 20 (1999), 424–431.
[5] R.A. Brualdi, Introductory combinatorics, 5th ed., Prentice Hall, Upper Saddle River, NJ, 2010.
[6] J.D. Camm, A.S. Raturi, and S. Tsubakitani, Cutting big M down to size. Interfaces 20 (1990), 61–66.
[7] D.W. Casbeer, A.L. Swindlehurst, and R. Beard, "Connectivity in a UAV multi-static radar network," in: Proc. American Institute of Aeronautics and Austronautics Guidance, Navigation, and Control Conference and Exhibit, Keystone, Colorado, 2006, pp. 1–8.
[8] T.M. Chan, Optimal output-sensitive convex hull algorithms in two and three dimensions, Discrete Comput Geom 16 (1996), 361–368.
[9] A.C. Coon, Spatial correlation of detections for impulsive echo ranging sonar, Johns Hopkins APL Tech Dig 18 (1997), 105–112.
[10] T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein, Introduction to Algorithms (3rd ed.), MIT Press, Cambridge, MA, 2009.
[11] H. Cox, "Fundamentals of bistatic active sonar," in: Y. Chan (Editor), Underwater acoustic data processing, Kluwer, Springer Netherlands, 1989, pp. 3–24.
[12] E.M. Craparo, J.P. How, and E. Modiano, Throughput optimization in mobile backbone networks, IEEE Trans Mob Comput 10 (2011), 560–572.
[13] E.M. Craparo and M. Karatas, Sensor optimization in multistatic underwater sensor networks, working paper, manuscript available on request (2016).
[14] M.P. Fewell and S. Ozols, Simple detection-performance analysis of multistatic sonar for anti-submarine warfare, Technical Report, DSTO-TR-2562, Defence Science and Technology Organisation, Edinburgh, South Australia, 2011.
[15] H. Freeman and R. Shapira, Determining the minimum-area encasing rectangle for an arbitrary closed curve, Commun ACM 18 (1975), 409–413.
[16] W.M. Gentleman and G. Sande, "Fast Fourier Transforms: For fun and profit," in: Proc. ACM Fall Joint Computer Conference, Boston, MA, 1966, pp. 563–578.
[17] C. George and D.R. DelBalzo, "Tactical planning with genetic algorithms for multi-static active sonobuoy systems," in: Proc. 19th International Congress on Acoustics, Madrid, Spain, 2007, pp. 1–6.
[18] X. Gong, J. Zhang, D. Cochran, and K. Xing, "Barrier coverage in bistatic radar sensor networks: Cassini oval sensing and optimal placement," in: Proc. 14th ACM International Symposium on Mobile Ad Hoc Networking and Computing, Bangalore, India, 2013, pp. 49–58.
[19] R.L. Graham, An efficient algorithm for determining the convex hull of a finite planar set, Inf Process Lett 1 (1972), 132–133.
[20] M. Gu and S.C. Eisenstat, A divide-and-conquer algorithm for the symmetric tridiagonal eigenproblem, SIAM J Matrix Anal Appl 16 (1995), 172–191.
[21] B.I. Incze and S.B. Dasinger, "A Bayesian method for managing uncertainties relating to distributed multistatic sensor search," in: Proc. 9th International Conference on Information Fusion, Florence, Italy, 2006, pp. 1–7.
[22] R.A. Jarvis, On the identification of the convex hull of a finite set of points in the plane, Inf Process Lett 2 (1973), 18–21.
[23] M. Kalkuhl, W. Wiechert, H. Nies, and O. Loffeld, "Simulation based optimization of bi-and multistatic SAR-missions," in: Proc. 7th European Conference on Synthetic Aperture Radar (EUSAR), Friedrichshafen, Germany, 2008, pp. 1–4.
[24] M. Kallay, The complexity of incremental convex hull algorithms in R^d, Inf Process Lett 19 (1984), 197.
[25] M. Karatas and E.M. Craparo, "Evaluating the direct blast effect in multistatic sonar networks using Monte Carlo simulation," in: Proc. IEEE Winter Simulation Conference (WSC), Huntington Beach, CA, USA, 2015, pp. 1184–1194.
[26] M. Karatas, M.M. Gunal, and E.M. Craparo, "Performance evaluation of mobile multistatic search operations via simulation," in: Proc. 49th Annual Simulation Symposium, Society for Computer Simulation International, Pasadena, CA, USA, 2016, pp. 110–115.
[27] A. Karatsuba and Y. Ofman, Multiplication of many-digital numbers by automatic computers, Dokl Akad Nauk SSSR 145 (1962), 293–294. Translat Phys Dokl 7 (1963), 595–596.
[28] D.G. Kirkpatrick, "Efficient computation of continuous skeletons," in: Proc. 20th Annual IEEE Symposium on Foundations of Computer Science, San Juan, Puerto Rico, 1979, pp. 18–27.
[29] D.G. Kirkpatrick and R. Seidel, The ultimate planar convex hull algorithm, SIAM J Comput 15 (1986), 287–299.
[30] D.E. Knuth, The art of computer programming: Sorting and searching, Vol. 3, Addison Wesley Publishing Company, Reading, MA, 1973.
[31] T.U. Kuhn, Optimal sensor placement in active multistatic sonar networks, Master's Thesis, Naval Postgraduate School, Monterey, CA, 2010.
[32] S.J. Leon, Linear algebra with applications, 8th ed., Prentice Hall, Upper Saddle River, NJ, 2010.
[33] G.L. Nemhauser and L.A. Wosley, An analysis of approximations for maximizing submodular set functions – I, Math Program 14 (1978), 265–294.
[34] P.N. Ngatchou, W.L. Fox, M. El-Sharkawi, "Multiobjective multistatic sonar sensor placement," in: Proc. IEEE Congress on Evolutionary Computation (CEC), Vancouver, BC, Canada, 2006, pp. 2713–2719.
[35] S. Ozols and M.P. Fewell, On the design of multistatic sonobuoy fields for area search, Technical Report, DSTO-TR-2563, Defence Science and Technology Organisation, Edinburgh, South Australia, 2011.
[36] F.P. Preparata and S.J. Hong, Convex hulls of finite sets of points in two and three dimensions, Commun ACM 20 (1977), 87–93.
[37] M.I. Shamos and D. Hoey, "Closest point problems," in: Proc. 16th Annual Institute of Electrical and Electronic Engineers Symposium on the Foundations of Computer Science, IEEE Computer Society, Berkeley, CA, 1975, pp. 151–162.
[38] S. Simakov, Localization in airborne multistatic sonars, IEEE J Ocean Eng 33 (2008), 278–288.
[39] C. Strode, "Optimising multistatic sensor locations using path planning and game theory," in: Proc. IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA), Paris, France, 2011, pp. 9–16.
[40] R. Tharmarasa, T. Kirubarajan, and T. Lang, "Joint path planning and sensor subset selection for multistatic sensor

networks," in: Proc. IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA), Ottawa, ON, Canada, 2009, pp. 1–8.

[41] Signal Online. 2017. http://www.afcea.org/content/?qtaxonomy/term/1258, AFCEA.

[42] M.J. Walsh and T. Wettergren, "Search performance prediction for multistatic sensor fields," in: Proc. IEEE Oceans Conference and Exhibition, Kobe, Japan, 2008, pp. 1–8.

[43] A.R. Washburn, A multistatic sonobuoy theory, Technical Report, NPS-OR-10-005, Naval Postgraduate School, Monterey, CA, 2010.

[44] A.R. Washburn and M. Karatas, Multistatic search theory, Military Oper Res 20 (2015), 21–38.