2021

Optimizing source and receiver placement in multistatic sonar

Craparo, Emily M.; Hof, Christoph; Fügenschuh, Armin; Karatas, Mumtaz

European Journal of Operational Research 272 (2019) 816–831

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Discrete Optimization

Optimizing source and receiver placement in multistatic sonar networks to monitor fixed targets

Emily M. Craparo[a,*], Armin Fügenschuh[b,1], Christoph Hof[c,2], Mumtaz Karatas[d]

[a] Department of Operations Research, Naval Postgraduate School, United States
[b] Helmut Schmidt University/University of the Federal Armed Forces Hamburg, Germany
[c] German Armed Forces and Studied Operations Research, The United States Naval Postgraduate School, Monterey, United States
[d] Industrial Engineering Department, National Defense University, Turkish Naval Academy, Istanbul, Turkey

ARTICLE INFO

Article history:
Received 21 December 2015
Accepted 1 February 2018
Available online 8 February 2018

Keywords:
OR in defense
Sensor placement
Optimization
Multistatic sonar

ABSTRACT

Multistatic sonar networks consisting of non-collocated sources and receivers are a promising development in sonar systems, but they present distinct mathematical challenges compared to the monostatic case in which each source is collocated with a receiver. This paper is the first to consider the optimal placement of both sources and receivers to monitor a given set of target locations. Prior publications have only considered optimal placement of one type of sensor, given a fixed placement of the other type. We first develop two integer linear programs capable of optimally placing both sources and receivers within a discrete set of locations. Although these models are capable of placing both sources and receivers to any degree of optimality desired by the user, their computation times may be unacceptably long for some applications. To address this issue, we then develop a two-step heuristic process, Adapt-LOC, that quickly selects positions for both sources and receivers, but with no guarantee of optimality. Based on this, we also create an iterative approach, Iter-LOC, which leads to a locally optimal placement of both sources and receivers, at the cost of larger computation times relative to Adapt-LOC. Finally, we perform computational experiments demonstrating that the newly developed algorithms constitute a powerful portfolio of tools, enabling the user to slect an appropriate level of solution quality, given the available time to perform computations. Our experiments include three real-world case studies.

Published by Elsevier B.V.

1. Introduction

Sonar systems have been in use in undersea and antisubmarine warfare for decades. During this time, they have evolved and found application in non-military fields, such as depth-finding, position marking, communication and telemetry, and aiding fishermen, divers, and conservationists (Urick, 1983). Researchers and practitioners distinguish between active and passive sonar systems as well as between monostatic and multistatic systems. A passive sonar system consists solely of receivers that "listen" for objects in the environment. An active sonar system contains at least one source and receiver. The source sends out a pulse of underwater sound, called a ping, which is reflected by objects in the underwater environment. The reflected signal is detected by a receiver, and using this signal it is possible to determine information about the objects in the vicinity, including their locations. A monostatic sonar system consists of sensors called posts; each post contains both a source and receiver (Ozols & Fewell, 2011). This principle is illustrated in Fig. 1 (left). In a multistatic sonar network (MSN),[3] sources and receivers are not necessarily collocated; see Fig. 1 (right). In the multistatic case, sources and receivers can consist of free-floating sonobuoys, or they can be mounted on ships or dipped by helicopters. A MSN has numerous advantages compared to a monostatic system. These advantages include reduced cost, more complicated countermeasures, increased flexibility, and higher precision with fewer pings. However, these advantages come at the cost of increased mathematical complexity in evaluating MSN system performance. The complications with MSNs arise due to the different geometry in comparison to the monostatic case. In the monostatic case the detection probability depends largely on the distance between post and target. In case of a multistatic constellation, the distances between target and source as well as target and receiver are relevant (Fewell & Ozols, 2011). As

* Corresponding author.
E-mail address: emcrapar@nps.edu (E.M. Craparo).
[1] Present address: Technical University Cottbus, Brandenburg, Germany.
[2] Studied Operations Research at the Naval Postgraduate School, United States.

[3] All acronyms used in this paper appear in Table 4 in the appendix.

https://doi.org/10.1016/j.ejor.2018.02.006
0377-2217/Published by Elsevier B.V.

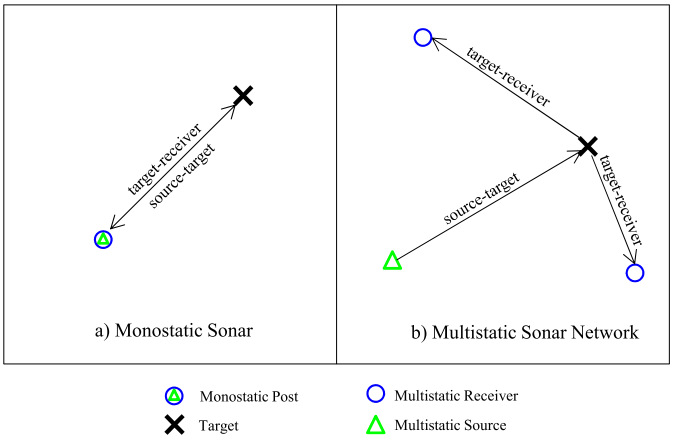


Fig. 1. Geometry of sonar detection for the monostatic case (left) and multistatic case (right).

a result, it is considerably more difficult to determine optimal positions for sources and receivers than for monostatic posts.

The problem of determining optimal positions for sensors in a MSN is also closely connected with facility location problems (FLPs), in particular with planar location or continuous facility location problems (CFLPs). In a CFLP, facilities are allowed to be located anywhere in an area of interest, as opposed to being restricted to a finite set of pre-identified locations (Arabani & Farahani, 2012; Carlo, Aldarondo, Saavedra, & Torres, 2012). Some problems that have been modeled as CFLPs include determining locations for Wi-Fi access points at airports, recreational areas, pollution sensors for environmental monitoring, and military surveillance devices (Revelle, Eiselt, & Daskin, 2008).

The literature on CFLPs is abundant and several models and solution techniques have been developed to address particular variants of the CFLP. For example, Redondo, Fernández, García, and Ortigosa (2009) study the CFLP problem in a competitive setting, where other facilities offering the same product or service already exist in the area. The authors solve the problem with three different heuristics approaches: a simulated annealing algorithm and two variants of evolutionary algorithms. Wong and Sun (2001) consider a heterogeneous continuous space with a set of competitive facilities and incorporate congested transportation costs between demand and facility nodes. They formulate the model as a combined distribution and assignment model and solve it with an iterative algorithm. On the other hand, Matisziw and Murray (2009) consider the problem of siting a facility in continuous non-convex space to maximize coverage. In contrast, Carlo et al. (2012) consider determining both the number and location of new facilities simultaneously with the objective of minimizing the total cost of interacting with a set of existing facilities. The researchers develop a nonlinear mixed integer mathematical model, a brute-force algorithm, and four heuristics, and they show that a greedy search heuristic outperforms all other heuristics considered. In a more recent study, Brimberg, Juel, Körner, and Schöbel (2015) study the CFLP under the assumption that a facility is allowed to cover a demand point partially. Although our problem can be considered to be a CFLP, it is clearly different from the other CFLP variants previously studied due to the presence of two types of "facilities" and the particular way these facilities interact in determining the objective value.

Despite the widespread use of multistatic sonar systems in practice, the literature contains relatively few analytical results to guide practitioners. Most of the existing studies present heuristic approaches or seek to evaluate a rule-of-thumb approach. For example, George and DelBalzo (2007) and Tharmarasa, Kirubarajan, and Lang (2009) use genetic algorithms to select locations for multistatic sensors for area coverage and tracking purposes. Ngatchou, Fox, El-Sharkawi et al. (2006) develop a particle swarm method to determine the number and placement of multistatic sensors to maximize area coverage. Similarly, Ozols and Fewell (2011) study the area coverage problem and analyze the coverage performance of 27 MSN layouts to determine the most cost effective pattern. Strode (2011) uses game theory to select multistatic sensor positions in order to detect a transiting intelligent underwater target; he then integrates this approach into the Multistatic Tactical Planning Aid (MSTPA), a decision support tool developed at the Centre for Maritime Research and Experimentation (CMRE). Kalkuhl, Wiechert, Nies, and Loffeld (2008) develop a simulation-based methodology for planning multistatic search and rescue missions. Casbeer, Swindlehurst, and Beard (2006) study the problem of connectivity in a mobile multistatic radar network containing unmanned air vehicles (UAVs). They develop a metric that provides for a balance between the performance and connectivity of the network. Gong, Zhang, Cochran, and Xing (2013) study another type of coverage problem: the barrier coverage problem, in which sensors are deployed on a line segment. They determine a placement order and spacing of sensors which minimizes the vulnerability of the network to intruders. Incze and Dasinger (2006) analyze 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 target behavior and target probability distribution. In another study, Bowen and Mitnick (1999) develop 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. Walsh, Wettergren et al. (2008) compute the expected detection probability of a given target track in a MSN field where all sources and receivers are distributed uniformly at random. Similarly, Washburn and Karatas (2015) 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. Karatas, Gunal, and Craparo (2016) use simulation to investigate the coverage performance of a mobile source performing parallel sweeps in a field of stationary receivers, and they compare their results by those of the analytic formulae developed by Washburn and Karatas (2015).

The aforementioned studies only consider the area coverage, barrier coverage, and tracking performance of a MSN. In contrast, Craparo and Karatas (2018) study the point coverage problem, in which the goal is to position sources in such a way as to cover as many of a finite number of target locations as possible, given fixed receiver locations. Their approach begins with a preprocessing algorithm that determines a polynomially-sized set of possible source locations guaranteed to contain the optimal source locations. Once this preprocessing is finished, source locations must be selected from among the set of candidate locations. Craparo and Karatas (2018) formulate an integer linear program that optimally selects source locations, and they also describe an efficient approximation algorithm for selecting source locations. In another study considering the point coverage problem, Craparo, Karatas, and Kuhn (2017) derive various results useful for excluding some suboptimal sensor locations, and they describe the Divide Best Sector (DiBS) algorithm for optimally placing a single source in a field of fixed receivers under a diffuse sensor model. In our computational experiments, we compare our algorithms' performance to that of Craparo and Karatas (2018), since their assumptions and overall problem setup most closely match our own.

Currently, no algorithm exists for selecting optimal sensor locations for both types of sensors (sources and receivers) for the point coverage problem. This capability is clearly desirable for practical multistatic operations, as in the case of a maritime patrol

aircraft deploying multistatic active (source) and passive (receiver) sonobuoys in an area of interest, with the goal of protecting the high value assets in the area. In this paper, we seek the optimal placement of a limited number of sources and receivers in a multistatic point coverage scenario. To accomplish this, we first develop an integer linear program, DISC-LOC-M, capable of optimally solving a discretized version of the problem. We then refine our model using an enumeration-based technique; this improves its computation time substantially. Our enumeration-based approach is denoted as DISC-LOC-enum. Although DISC-LOC-enum's computation time is preferable to that of DISC-LOC-M, it nevertheless may be unacceptably long for some applications. Thus, we develop two heuristic algorithms: Adapt-LOC performs a two-step process to very rapidly determine positions for both sources and receivers, while Iter-LOC extends the approach of Adapt-LOC to generate a locally optimal placement of sources and receivers.

2. Assumptions and preliminaries

Our goal is to place a limited number of multistatic sonobuoys so as to observe as many target locations as possible from a given set T. We have available a set of sources, S and a set of receivers, R. For simplicity, we assume the following:

- Targets are stationary points that we wish to observe, and their locations are specified.
- The sensors have definite range ("cookie-cutter") detection curves, with perfect detection within the curve and no detection outside.
- The range of the day (RoD, ρ_0) is a known value representing the maximum distance from which a monostatic sensor could detect a target.
- The direct blast effect is negligible (see Cox, 1989; Fewell & Ozols, 2011; Karatas & Craparo, 2015 for details).

2.1. Multistatic detection theory

Detection in a monostatic sonar system depends on the distance between the target and the sensor, d. Under our assumptions, the probability of detection $p_{detection}$ is given by Eq. (1). (For additional background on detection theory and mathematical deduction of this equation, see Craparo & Karatas, 2018; Urick, 1983; Washburn & Karatas, 2015, or Ozols & Fewell, 2011.)

$$p_{\text{detection}} = \begin{cases} 1 & \text{if } d \le \rho_0, \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

In the multistatic case, the geometry of the detection problem becomes more complex, as shown in Fig. 2 for a configuration with two sources, two targets, and three receivers.

The probability that source s and receiver r will detect target t, $p_{t,s,r}$, depends on the distance between target and source ($d_{t,s}$) as well as target and receiver ($d_{t,r}$) and can be calculated via Eq. (2),

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

Multiple sources and receivers result in different combinations of the distances to detect a target. However, note that for a definite range sensor model, it is sufficient that the product of the two shortest distances fulfill the condition in Eq. (2).

3. Enhanced sensor placement

We now describe new approaches for placing multistatic sources and receivers.

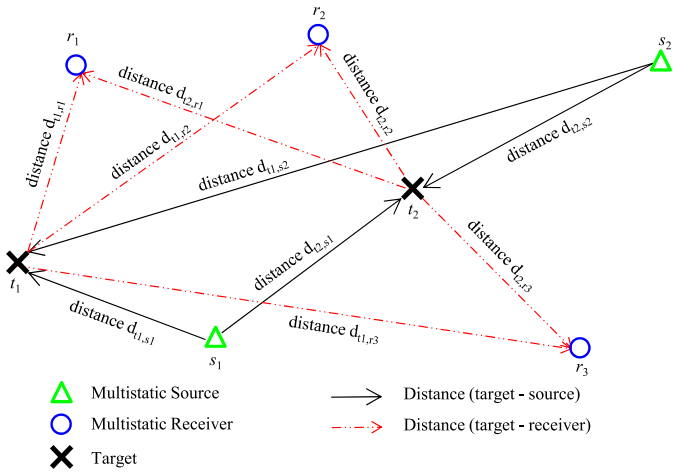


Fig. 2. Geometry of multistatic sonar detection: the signal is sent by the sources (s_1 and s_2), reflected by the targets (t_1 and t_2) and sensed by the receivers (r_1, r_2 and r_3).

3.1. Optimized sensor placement using mathematical programming

Our first two approaches consist of mathematical programming models. Both models select optimal locations for sources and receivers from within a discrete set of candidate locations G. This set of candidate locations may be derived using operational requirements, or it may simply represent discretization of the target space using a uniform grid structure. Each model provides an exact solution to this discrete approximation of the sensor placement problem. This approximation can be made arbitrarily good by increasing the number of candidate locations for sensors, at the cost of increased computational effort.

The two models differ in how they compute the coverage achieved by a given sensor placement. Our first model, DISC-LOC-M, uses a big-M approach to determining target coverage. Our second model, DISC-LOC-enum, uses an enumeration-based approach instead. We conclude this section by conducting computational experiments to compare the two approaches.

3.1.1. DISC-LOC-M

DISC-LOC-M linearizes Eq. (2) by taking the logarithm of both sides of the detection requirement. Where before we had

$$p_{t,s,r} = \begin{cases} 1 & \text{if } d_{t,s}d_{t,r} \le \rho_0^2, \\ 0 & \text{otherwise,} \end{cases} \tag{3}$$

we now state that

$$p_{t,s,r} = \begin{cases} 1 & \text{if } \tilde{d}_{t,s} + \tilde{d}_{t,r} \le 2\tilde{\rho}_0, \\ 0 & \text{otherwise,} \end{cases} \tag{4}$$

where $\tilde{d}_{t,s}$, $\tilde{d}_{t,r}$, and $\tilde{\rho}_0$ are the log-transformed versions of $d_{t,s}$, $d_{t,r}$, and ρ_0:

$$\tilde{d}_{t,s} = \log d_{t,s}$$
$$\tilde{d}_{t,r} = \log d_{t,r}$$
$$\tilde{\rho}_0 = \log \rho_0.$$

This linearization yields the following model:

Model DISC-LOC-M:

Sets and indices:

$g, g' \in G$: set of candidate locations for sensors
$t \in T$: set of targets

Parameters:

$M_{t,g,g'}$: a large number used for relaxing a constraint
$\tilde{d}_{g,t}$: log-transformed distance between candidate location g and target t
$|R|$: number of receivers
$|S|$: number of sources
$\tilde{\rho}_0$: log-transformed range of the day.

Binary variables:

s_g : 1 if a source is located on candidate location g
r_g : 1 if a receiver is located on candidate location g
$h_{t,g,g'}$: 1 if target t is covered by a source located at candidate location g and a receiver located at candidate location g'
c_t : 1 if target t is covered by at least one pair of source and receiver

Formulation:

$$\max z = \sum_{t \in T} c_t \tag{5}$$

s.t.

$$\tilde{d}_{g,t} s_g + \tilde{d}_{g',t} r_{g'} \leq 2\tilde{\rho}_0 + \left(1 - h_{t,g,g'}\right) M_{t,g,g'} \quad \forall t \in T, \forall g, g' \in G \tag{6}$$

$$h_{t,g,g'} \leq s_g \quad \forall t \in T, \forall g, g' \in G \tag{7}$$

$$h_{t,g,g'} \leq r_{g'} \quad \forall t \in T, \forall g, g' \in G \tag{8}$$

$$c_t \leq \sum_{g \in G} \sum_{g' \in G} h_{t,g,g'} \quad \forall t \in T \tag{9}$$

$$|S| \geq \sum_{g \in G} s_g \tag{10}$$

$$|R| \geq \sum_{g \in G} r_g \tag{11}$$

$$s_g, r_g \in \{0, 1\} \quad \forall g \in G \tag{12}$$

$$c_t \in \{0, 1\} \quad \forall t \in T \tag{13}$$

$$h_{t,g,g'} \in \{0, 1\} \quad \forall t \in T, \forall g, g' \in G \tag{14}$$

The objective function (5) expresses the total number of targets covered. Constraints (6), (7), and (8) ensure that the multistatic detection criterion is satisfied if target t is covered ($h_{t,g,g'} = 1$) by a source and receiver located on candidate locations g and g', respectively. If target t is not covered ($h_{t,g,g'} = 0$), then a large number M is added to the right hand side of the constraint (6), thus relaxing it. Constraint (9) ensures that each target t is covered only if the detection criterion is met for at least one pair of grid points. Constraints (10) and (11) ensure that the total allocated sensors do not exceed the sensors available. Constraints (12)–(14) declare variable types.

3.1.2. DISC-LOC-enum

As an alternative to DISC-LOC-M's big-M approach, DISC-LOC-enum performs a preprocessing step to evaluate which pairs of grid points $(g, g') \in G \times G$ are capable of covering each target t when populated with a source and receiver. By performing this evaluation in preprocessing, we avoid the need to evaluate Eq. (2) within the model. This can substantially reduce computation time, since a big-M approach often leads to poor linear program relaxations.

Model DISC-LOC-enum:
Sets and indices:

$g, g' \in G$: set of candidate locations for sensors
$t \in T$: set of targets
$(t, g, g') \in D$: set of triples such that target t is detectable by a source and receiver placed at locations g and g': $D := \{(t, g, g') \in T \times G \times G | P_{t,g,g'} = 1\}$

Parameters:

$|S|$: number of sources available
$|R|$: number of receivers available

Binary variables:

s_g : 1 if a source is located in candidate location g
r_g : 1 if a receiver is located in candidate location g
$h_{g,g'}$: 1 if a source is located at candidate location g and a receiver is located at candidate location g'
c_t : 1 if target t is covered by at least one pair of source and receiver

Formulation:

$$\max z = \sum_{t \in T} c_t \tag{15}$$

s.t.

$$h_{g,g'} \leq s_g \quad \forall g, g' \in G \tag{16}$$

$$h_{g,g'} \leq r_{g'} \quad \forall g, g' \in G \tag{17}$$

$$c_t \leq \sum_{(g,g'):(t,g,g') \in D} h_{g,g'} \quad \forall t \in T \tag{18}$$

$$|S| \geq \sum_{g \in G} s_g \tag{19}$$

$$|R| \geq \sum_{g \in G} r_g \tag{20}$$

$$s_g, r_g \in \{0, 1\} \quad \forall g \in G \tag{21}$$

$$c_t \in \{0, 1\} \quad \forall t \in T \tag{22}$$

$$h_{g,g'} \in \{0, 1\} \quad \forall g, g' \in G \tag{23}$$

The objective function (15) expresses the total number of targets covered. Constraints (16) and (17) ensure that the multistatic detection criterion is satisfied if target t is covered by a source and receiver located on candidate locations g and g', respectively. Constraint (18) ensures that each target t is covered only if the detection criterion is met for at least one pair of grid points. Constraints (19) and (20) ensure that the total allocated sensors do not exceed the sensors available. Constraints (21)–(23) declare variable types.

3.1.3. Computational performance of DISC-LOC-M and DISC-LOC-enum

To compare the performance of DISC-LOC-M and DISC-LOC-enum, we perform computational experiments on problems of varying sizes. For simplicity, each of our problem instances consists of a uniform grid of points, where each point acts as both a target to be covered and a candidate sensor location. We consider a number of problem sizes and sensor mixes, and we record the time taken for DISC-LOC-M and DISC-LOC-enum to solve each instance. (Because both models solve the same problem to proven global optimality, they achieve identical coverage performance. We compare this coverage performance to that of other approaches in Section 4.) Fig. 3 displays the results of our experiments. As the figure indicates, DISC-LOC-M requires substantially more computation time across all problem sizes and sensor mixes. On average, DISC-LOC-M requires 15.7 times as much computation time as DISC-LOC-enum.

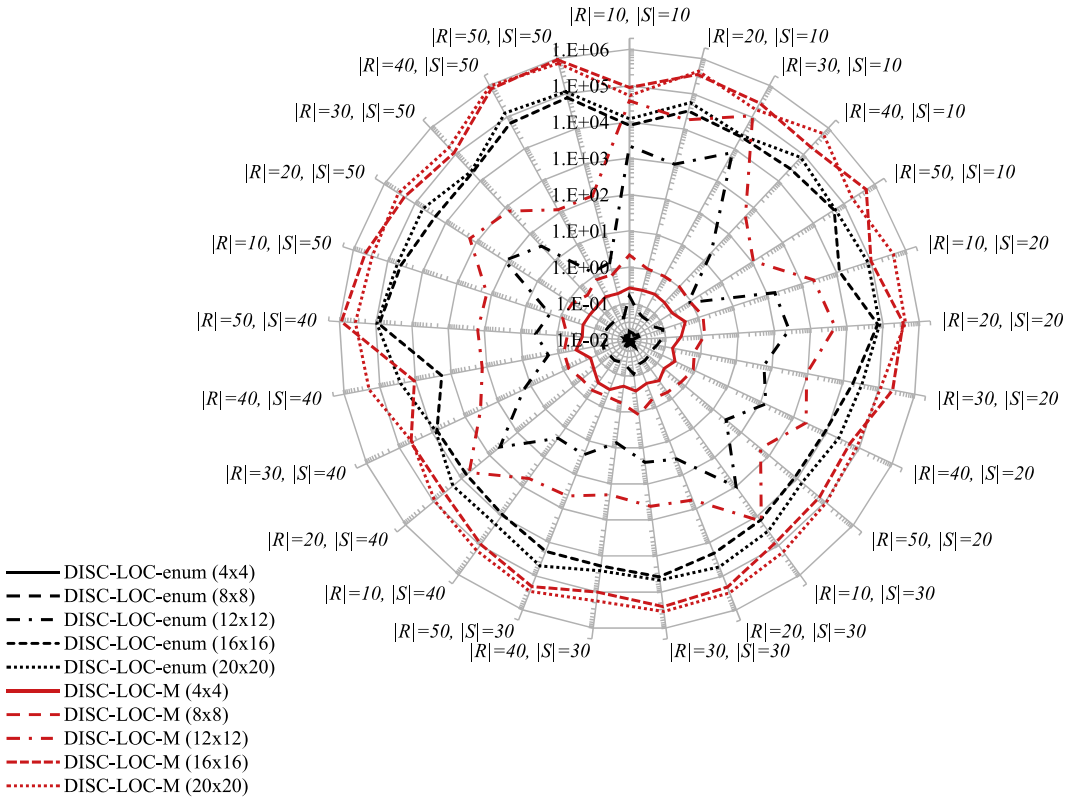


Fig. 3. Computational performance of DISC-LOC-M and DISC-LOC-enum, for varying problem instances. In each instance, we consider a uniform grid of points, where each point is both a candidate sensor location and a target to be covered. The grid dimensions are indicated in the figure legend. For each grid configuration, we consider a variety of sensor mixes, as indicated on the circumferential axis. The radial axis indicates the computation time in seconds.

Although DISC-LOC-enum substantially outperforms DISC-LOC-M, its computation times may nevertheless be too long for practical applications. Thus, we now turn our attention to the development of heuristic algorithms capable of providing solutions more quickly than DISC-LOC-enum for large problem instances, albeit without guaranteed optimality. For the remainder of this paper, we consider only DISC-LOC-enum in our computational experiments, and we denote it simply as DISC-LOC.

3.2. Detection discs

In order to develop our heuristic approaches, we build upon prior work that only considers optimal placement of sources in a network of fixed targets and receivers. We also utilize geometric constructs known as *detection discs*. The concept of detection discs in multistatic systems is first introduced by Craparo and Karatas (2018). In general, a detection disc for a particular target is the region in which a sensor can be placed that will detect that target. For the monostatic case, the boundary of a detection disc consists of a circle around the target with radius ρ_0. A monostatic sensor placed inside that region will detect the target. Regions in which multiple detection discs overlap represent locations where one sensor is able to detect more than one target. In Fig. 4, for example, three detection discs overlap at point p_2, and so one monostatic sensor at p_2 detects targets t_1, t_2, and t_3. To detect all of the rightmost group of targets, one could place sensors at any two of the three locations in the set $\{p_5, p_6, p_7\}$. Each of the points in the set $\{p_1, p_2, p_3, p_4, p_5, p_6, p_7\}$ covers a maximal set of targets that can be detected by a single monostatic sensor, meaning that it is not possible to cover all of the relevant targets as well as at least one other target with a single sensor. Following Craparo and Karatas (2018), we refer to the discs that overlap at these points as *maximal sets of mutually overlapping detection discs*.

For the multistatic case in which receivers are already placed, the detection disc for a target consists of those locations where a source can be placed that will detect that target. Given a target t and receiver r separated by distance $d_{t,r}$, a source placed within radius $\frac{\rho_0^2}{d_{t,r}}$ of the target allows detection. Since only one source–receiver pair is required for detection in a definite range sensor model, it is sufficient to consider only the closest receiver to each target. Thus, we obtain the probability of detecting target t, p_t, as

$$p_t = \begin{cases} 1 & \text{if } d_{t,s^*(t)} \le \rho_0^2 / d_{t,r^*(t)} \\ 0 & \text{otherwise} \end{cases} \qquad \forall t \in T, \tag{24}$$

where $d_{t,s^*(t)}$ is the distance from target t to its nearest source, and $d_{t,r^*(t)}$ is the distance from target t to its nearest receiver (Craparo & Karatas, 2018).

3.3. Preprocessing and optimized source placement using detection discs

For the following, we assume a setting with known positions of a set of targets $t \in T$ and receivers $r \in R$. Targets and receivers occupy a square, two dimensional area of interest. Eq. (24) implies the existence of circular detection discs around each target; we denote target t's disc as δ_t. The detection discs comprise a set $D = \{\delta_1, \delta_2, \ldots, \delta_{|T|}\}$, and placement of sensors in overlapping regions of these discs can lead to detection of multiple targets with one sensor. Using this insight, Craparo and Karatas (2018) determine optimal source positions by the following process:

- Use detection discs to identify maximal subsets of targets detectable by a single source.

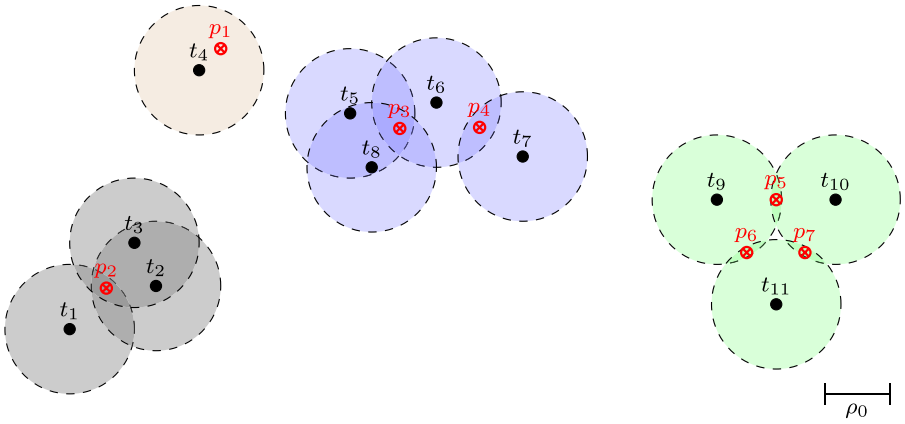


Fig. 4. *Detection disc example* – detection discs for a set of several targets when using a monostatic sensor with detection radius ρ_0. .

0. Input parameters are: set of receivers R with positions $x_r, \forall r \in R$, set of targets T with positions $x_t, \forall t \in T$, RoD ρ_0
1. Compute distances between targets and receivers $d_{t,r}, \forall t \in T, r \in R$, find $d_{t,r^*(t)} = \min_r d_{t,r}, \forall t \in T$, determine distances between all targets $d_{t,t'}, \forall t, t' \in T$
2. Create detection discs with $\delta_t \in D$, each with center x_t and radius $\rho_0^2/d_{t,r^*(t)}, \forall t \in T$
3. For all pairs of detection discs $\delta_t, \delta_{t'} \in D$ calculate all intersection points $I_{t,t'}$ and build the set of all intersection points $\overline{I} = \bigcup_{t,t' \in T} I_{t,t'}$, where each $I_{t,t'}$ can have between zero and two entries
4. Generate the set of all possible candidate location points $C = \{x_1, x_2, ..., x_T\} \bigcup \overline{I}$
5. Generate the reduced candidate point set by eliminating each point in C that does not represent a maximal set of mutually overlapping detection discs

Fig. 5. Preprocessing algorithm LOC-GEN, which determines a polynomially sized set of candidate source locations C guaranteed to contain the optimal source locations (Craparo & Karatas, 2018).

- For each subset of targets, determine a location in which a source could be placed and detect those targets.
- From among the resulting set of candidate locations, determine the best subset of location to place a limited number of available sources.

The first two steps can be thought of as preprocessing steps, while the third involves optimization. This process is similar to that proposed by Church (1984) and later improved by He, Fan, Cheng, Wang, and Tan (2016) in the context of the planar maximal covering location problem (PMCLP). In particular, the algorithm developed by Church (1984) considers locating facilities on a two-dimensional continuous plane to cover a given set of demand points. The author shows that the circle intersect point set (CIPS), a set which consists of all demand points plus all points in which some pair of circles intersect, is guaranteed to contain an optimal set of facility locations. He et al. (2016) further develop the CIPS concept and propose a deterministic algorithm, called the mean-shift based algorithm, for solving large-scale PMCLPs.

Craparo and Karatas (2018) develop a polynomial-time preprocessing algorithm called LOC-GEN, the pseudo-code of which is depicted in Fig. 5, to generate candidate source locations. This set of candidate locations is guaranteed to contain an optimal set of source locations. They then develop two different approaches to select among these locations. One is an integer linear program named OPT-LOC, and the other is a polynomial-time approximation algorithm based on a greedy approach (Greedy-LOC). The pseudo-code of both algorithms appears in Craparo and Karatas (2018). The overall procedure allows the optimal or near-optimal placement of sources for a given set of targets and receivers. In Hof (2015), we enhance Craparo and Karatas's (2018)) LOC-GEN algorithm to improve its computational efficiency. Craparo and Karatas (2018) leverage the fact that optimal source locations are guaranteed to be contained in the set of all detection disc center points and the intersection points of the boundaries of detection discs. LOC-GEN then proceeds to eliminate any such points that are "dominated," in the sense that at least one other point lies at the intersection of a superset of detection discs. Our new algorithm, LOC-GEN-II (see Fig. 6), improves computational efficiency in part by not considering the center locations of any detection disc that has at least one intersection point. The motivation for this modification is described in Lemma 1. LOC-GEN-II further improves LOC-GEN via various implementation techniques. For more information about LOC-GEN-II, see Hof (2015).

Lemma 1. *For each target t whose detection disc boundary has at least one intersection point with another target's detection disc boundary, the center point of the detection disc δ_t can be discarded as an optimal source location. The center point of a detection disc need only be included in the set of candidate source locations if the detection disc has no intersections.*

Proof. Consider a maximal set of mutually overlapping detection discs $\overline{D}_0$ that covers region $R(\overline{D}_0)$. As noted by Craparo and Karatas (2018), the boundary of region $R(\overline{D}_0)$ consists of portions of the boundaries of detection discs in $\overline{D}_0$. Suppose the boundary of

0. Input parameters are: receivers R, $x_r, \forall r \in R$, targets T, $x_t, \forall t \in T$, RoD ρ_0.
1. Compute $d_{t,r}, \forall t \in T, r \in R$, find $d_{t,r^*(t)} = \min_r d_{t,r}, \forall t \in T$, determine $d_{t,t'}, \forall t, t' \in T$.
2. Create detection discs with $\delta_t \in D$, each with center x_t and radius $\Delta_t = \rho_0^2/d_{t,r^*(t)}, \forall t \in T$.
3. For all pairs of detection discs $\delta_t, \delta_{t'} \in D$ calculate all $I^k_{t,t'}$ and build $\overline{I} = \bigcup_{t,t' \in T,k} I^k_{t,t'}$, where k refers to the k^{th} intersection point of the detection discs of targets t and t'.
4. For each intersection point $I^k_{t,t'}$, determine the targets that can be detected from this location, $\tau(I^k_{t,t'})$.
5. Build the center point list $\overline{P} = \bigcup_{t \in T} x_t$.
6. If a targets detection disc has at least one intersection point, erase its center location from the list of center points $\overline{P}$.
7. Generate the final reduced candidate point list $\overline{C^\star}$ by eliminating points in $\overline{P}$ and $\overline{I}$ which do not represent a maximal set of mutually overlapping detection discs by the following process:

   Step 1. For each pair of center points x_i and x_j, $i \neq j$ calculate the distance between them $dist(i,j)$. If $dist(i,j) < \Delta_i$ and $dist(i,j) > \Delta_j$ erase x_i from $\overline{P}$. Otherwise, leave x_i in $\overline{P}$.

   Step 2. Set $\overline{C^\star} = \overline{P}$.

   Step 3. While $\overline{I} \neq \emptyset$ find an intersection point that covers as many targets as possible. That is, find a point $I^k_{t,t'} \in \overline{I}$ such that $\left|\tau(I^k_{t,t'})\right| = \max_{\tau(I^k_{t^n,t^m}) \in \overline{I}} \left|\tau(I^k_{t^n,t^m})\right|$. (Break ties arbitrarily.) Then, for each $I^k_{t^n,t^m}$ such that $\tau(I^k_{t^n,t^m}) \subseteq \tau(I^k_{t^n,t^m})$ erase $I^k_{t^n,t^m}$ from $\overline{I}$. Finally, add $I^k_{t^n,t^m}$ to $\overline{C^\star}$ and erase it from $\overline{I}$.
8. Return $\overline{C^\star}$.

Fig. 6. Improved preprocessing algorithm LOC-GEN-II.

region $R(\overline{D}_0)$ consists of the boundary of a single detection disc δ_t. In this case, the boundary of disc δ_t cannot intersect another detection disc $\delta_{t'}$: such an intersection would imply that some areas of the boundary are covered by $\delta_{t'}$ while others are not, contradicting our statement that the boundary of detection disc δ_t encloses a region $R(\overline{D}_0)$ of mutually overlapping detection discs. As noted by Craparo and Karatas (2018), the center point of δ_t is sufficient to characterize region $R(\overline{D}_0)$.

Suppose now that the boundary of $R(\overline{D}_0)$ consists of portions of the boundaries of multiple detection discs. Craparo and Karatas (2018) prove that in this case there always exist intersection points $I_{t,t'} \in R(\overline{D}_0)$, and thus $R(\overline{D}_0)$ need not be represented by a center point. Thus, we need only consider center points of detection discs that do not intersect with other detection discs, as such points may (but need not) be required in order to characterize regions whose boundaries consist of the boundaries of single detection discs. □

3.4. Heuristic approaches for sensor placement

We now extend the work of Craparo and Karatas (2018) by developing two approaches for placing receivers and sources when the locations of all targets are given. The pseudo-code for our first approach, named Adapt-LOC, appears in Fig. 7.

Initially, only the target positions are given. Thus, we cannot directly use the approach of Craparo and Karatas (2018) in determining detection discs, since receiver locations are unknown. As a heuristic approach, we instead create a disc of fixed radius $\alpha\rho_0$ around each target, where α is a scalar of our choosing. Given these discs, we then proceed with LOC-GEN-II as before, creating a set of candidate locations for receivers by identifying maximal sets of mutually overlapping discs of radius $\alpha\rho_0$. We use OPT-LOC or Greedy-LOC to determine the $|R|$ best of these locations. With the receivers placed, we can now use a combination of LOC-GEN-II and OPT-LOC or Greedy-LOC to determine the optimal places for $|S|$ sources, given our heuristic placement of the receivers.

Fig. 8 provides a comparison between a solution produced by Adapt-LOC and one produced by the algorithm of Craparo and Karatas (2018), modified to include random placement of receivers. (This modified version of the algorithm of Craparo and Karatas (2018) henceforth be referred to as the C&K algorithm.) Qualitative differences between the solutions produced by Adapt-LOC and the C&K algorithm, modified to include random placement of receivers, can be seen in Fig. 8. Given is a set of targets (×). In the leftmost part of the figure we see the candidate positions (●) and the best locations (○) for the receivers as chosen by Adapt-LOC. The middle plot shows the selected positions of the receivers (○), the candidate points for the positions of the sources determined by LOC-GEN-II (●), and the optimally placed sources (△). The right plot shows the targets (×) and randomly placed receivers (○) and, for this configuration, the results for the optimal places of the sources (● and △). Adapt-LOC detects 80% of the targets in this example, compared to 50% for the C&K algorithm. Looking at the qualitative structure of the solutions, we can see that Adapt-LOC provides a clustering of sensors in those areas in which targets are most densely located, in contrast to the C&K algorithm with randomly placed receivers.

But Adapt-LOC also has disadvantages. For instance, its detection performance is strongly dependent on the chosen value for α, as well as the number of sources and receivers. Tests for

0. Input parameters are: targets T, locations $x_t, \forall t \in T$, RoD ρ_0, scalar α, number of receivers $|R|$, and number of sources $|S|$
1. Create detection discs $\delta_t \in D$, with center x_t and radius $\alpha\rho_0, \forall t \in T$
2. For all pairs of detection discs $\delta_t, \delta_{t'} \in D$ calculate all $I_{t,t'}$ and build $\overline{I} = \bigcup_{t,t' \in T} I_{t,t'}$
3. Execute LOC-GEN-II
4. Choose $|R|$ receiver locations using OPT-LOC or Greedy-LOC
5. Execute LOC-GEN-II with the given sets of targets and receivers (recompute detection discs with centers x_t and radii $\frac{\rho_0^2}{d_{t,r^*(t)}}$)
6. Choose the $|S|$ source locations using OPT-LOC or Greedy-LOC

Fig. 7. Adapt-LOC algorithm for selecting locations for sources and receivers given a set of targets.

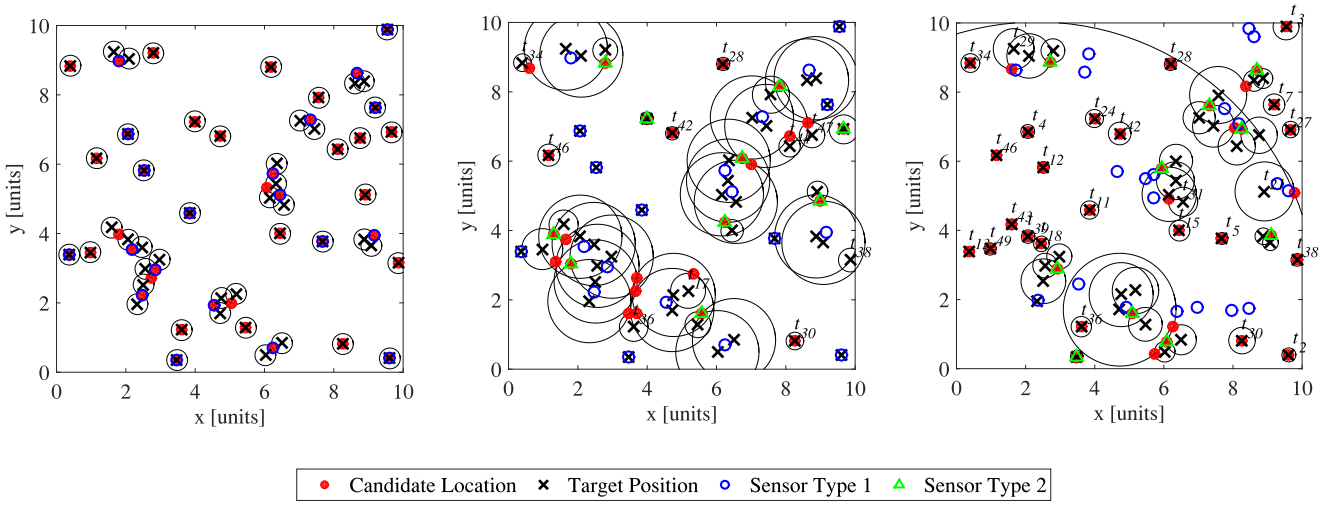


Fig. 8. Sensor placement using Adapt-LOC and the C&K algorithm: the left and middle figures show the solution selected by Adapt-LOC for $|T| = 50$ targets, $|R| = 20$ receivers, and $|S| = 10$ sources. The leftmost figure shows the receiver positions selected by Adapt-LOC using $\alpha = 0.5\rho_0$, while the middle figure shows the resulting sources positions. The rightmost figure shows the sensor layout resulting from random placement of the receivers and near-optimal placement of sources. In all cases, we use Greedy-LOC to select among candidate locations. In this particular example, Adapt-LOC allows detection of 80% of all targets, while the C&K algorithm detects 50%. Undetected targets are labeled with their number t_n.

Table 1
Relationship between number of sensors, radius of target detection discs, and detection rate.

# receivers $\|R\|$	# sources $\|S\|$	Radius of target detection discs α	Detection rate $\#t_{\text{detected}}/\#t$
High	Low	Low	Very high
Low	High	Low	Low
High	Low	High	Very low
Low	High	High	High

different configurations of the parameters show that values of $0.25 \leq \alpha \leq 0.75$ lead to the best results; the overall dependencies are as shown in Table 1.

To address Adapt-LOC's sensitivity to α, we extend Adapt-LOC to produce locally optimal solutions using an iterative approach. We denote this development as Iter-LOC; Fig. 9 shows its pseudocode. As before, initially, only the positions of the targets are known. In the first step, Iter-LOC determines receiver positions for some value of α, as in Adapt-LOC. In the second step, Iter-LOC determines positions for the sources based on the current target and receiver positions. If OPT-LOC is used as a subroutine for selecting source positions, these positions are optimal. Otherwise, if Greedy-LOC is used (for instance), they may be suboptimal. In the third step, Iter-LOC determines receiver positions given the current target and source positions, and it proceeds to iterate in this manner, successively fixing either sources or receivers and determining locations for the other sensor type. This process can be repeated until no further improvement is achieved (if OPT-LOC is used to select sensor locations), or until the improvement is sufficiently small. Using OPT-LOC as a subroutine results in a locally optimal placement of receivers and sources.

4. Results

We now perform computational experiments to demonstrate the benefits of our proposed algorithms. First, we perform randomized trials comparing our algorithms to other approaches from the literature. Then, we demonstrate the applicability of our algorithms to practical instances. In all experiments, we use DISC-LOC-enum rather than DISC-LOC-M, and we denote it simply as DISC-LOC.

4.1. Randomized experiments

We now describe the results of computational experiments comparing DISC-LOC, Adapt-LOC, and Iter-LOC with the C&K algorithm and the algorithm of Washburn and Karatas (2015), henceforth referred to as the W&K algorithm. (Recall that

0. Input parameters are: targets T, $x_t, \forall t \in T$, RoD ρ_0, $\alpha \in [0.25, 0.75]$, ε (threshold for detection performance improvement), number of receivers $|R|$, number of sources $|S|$
1. Execute Adapt-LOC
2. Revise $|R|$ receiver locations using OPT-LOC or Greedy-LOC

   If termination condition is met **then**,

   STOP

   else

   go to Step 3
3. Revise $|S|$ source locations using OPT-LOC or Greedy-LOC

   If termination condition is met **then**,

   STOP

   else

   go to Step 2

**Example termination conditions:*

- Improvement in detection performance $= 0$ for OPT-LOC
- Improvement in detection performance $\leq \varepsilon$ for Greedy-LOC

Fig. 9. Iter-LOC algorithm for placing sources and receivers given a set of targets.

Washburn & Karatas, 2015 consider uniform random placement of both sources and receivers.) We use the following computer configuration: AMD K16 Mullins Processor (A4-6210) with four kernels and 1800 megahertz clock frequency per kernel, 100 megahertz BUS clock frequency, operating system Microsoft Windows 8.1 64-Bit, 6144 megabytes DDR3 RAM with clock frequency 798.4 megahertz. Software is MATLAB R2014b, version 8.4.0.150421, and GAMS Release 24.2.1 r43572 and CPLEX version 12.2.0.2 for x86_64/MS Windows. For each experiment and parameter configuration, we compute 100 replications with 200 targets randomly placed in a 10×10 unit 2-dimensional area, and we fix $\rho_0 = 0.6$ units. We consider various sensor mixes, reflected by the number of receivers ($|R|$) and sources ($|S|$). This configuration is similar to the setup considered by Craparo and Karatas (2018).

Fig. 10 examines the performance of DISC-LOC at varying levels of resolution. The goal is to maximize the number of targets detected; a value of one reflects detection of all 200 targets. The top portion of the figure shows the average fraction of targets detected, while the bottom shows the average computation time for each sensor mix. Note that although DISC-LOC solves very quickly at a resolution of 4×4, its detection performance is quite poor at that resolution. At a resolution of 20×20, we obtain excellent detection performance at the cost of high computation times. The computation time results presented in Fig. 10 do not include preprocessing time; the average preprocessing time ranged from 0.009 seconds for a 4×4 resolution to 17 seconds for a 20×20 resolution.

In Fig. 11, we compare the detection performance of DISC-LOC at its highest resolution (20×20) with Adapt-LOC, Iter-LOC, the C&K algorithm, and the W&K algorithm.

A few main trends are apparent:

1. DISC-LOC's detection performance exceeds that of all other approaches at a resolution of 20×20.
2. Pure random sensor placement is always outperformed by the C&K algorithm, which is in turn outperformed by Adapt-LOC for all chosen values of α.
3. We observe variations in Adapt-LOC's performance with varying α. The best performance for our instances occurs with $\alpha \approx 0.5$; this setting results in approximatly 2%–3% better detection performance than the other setting. Iter-LOC's performance is nearly identical for all values of α, and thus we show only one value in the figure.

The results of this experiment indicate that Adapt-LOC allows up to 25% higher detection rates compared to the C&K algorithm. This is a significant performance increase. On the other hand, Iter-LOC outperforms Adapt-LOC, and DISC-LOC outperforms all other algorithms at its highest resolution. Iter-LOC increases the detection rate by up to 13% compared to Adapt-LOC. Thus, Iter-LOC increases the maximum performance gain in comparison to the C&K algorithm to 38%. DISC-LOC, in turn, outperforms Iter-LOC by an average of 2% higher detection rate. The maximum difference is 7% when $|R| = 30,\ |S| = 20$. We also note that, for our problem instances, an increase in the number of sensors leads nearly to the same gain in detection performance, regardless of sensor type. For example, take the starting point $|R| = 10,\ |S| = 10$ with a detection probability of approximately 42%. An additional 10 receivers ($|R| = 20,\ |S| = 10$) lead to a detection probability of nearly 57%, while 10 additional sources ($|R| = 10,\ |S| = 20$) lead to approximately 56% detection probability. If this relation holds for all configurations, it has fortunate implications for the practical aspects of fielding MSNs. Using Iter-LOC different with values of $\alpha \in \{0.25, 0.5, 0.75\}$ results in nearly identical detection performance. This implies that, as desired, the starting value of α has only a negligible impact on the overall detection of targets when using Iter-LOC.

To investigate the effect of the different sensor placement subroutines, we compare the performance achieved when using the Greedy-LOC and OPT-LOC subroutines. The outcomes of these experiments appear in Fig. 12. We see an average increase in detection rate of around 5% using Iter-LOC as opposed to Adapt-LOC, regardless of whether Greedy-LOC or OPT-LOC is used. Thus, for our problem instances, the use of OPT-LOC only leads to minor improvement of detection probability, while the computational effort per iteration is high. However, we also observe that the performance of Greedy-LOC may not improve monotonically with the iteration steps. This is because Greedy-LOC does not deliver

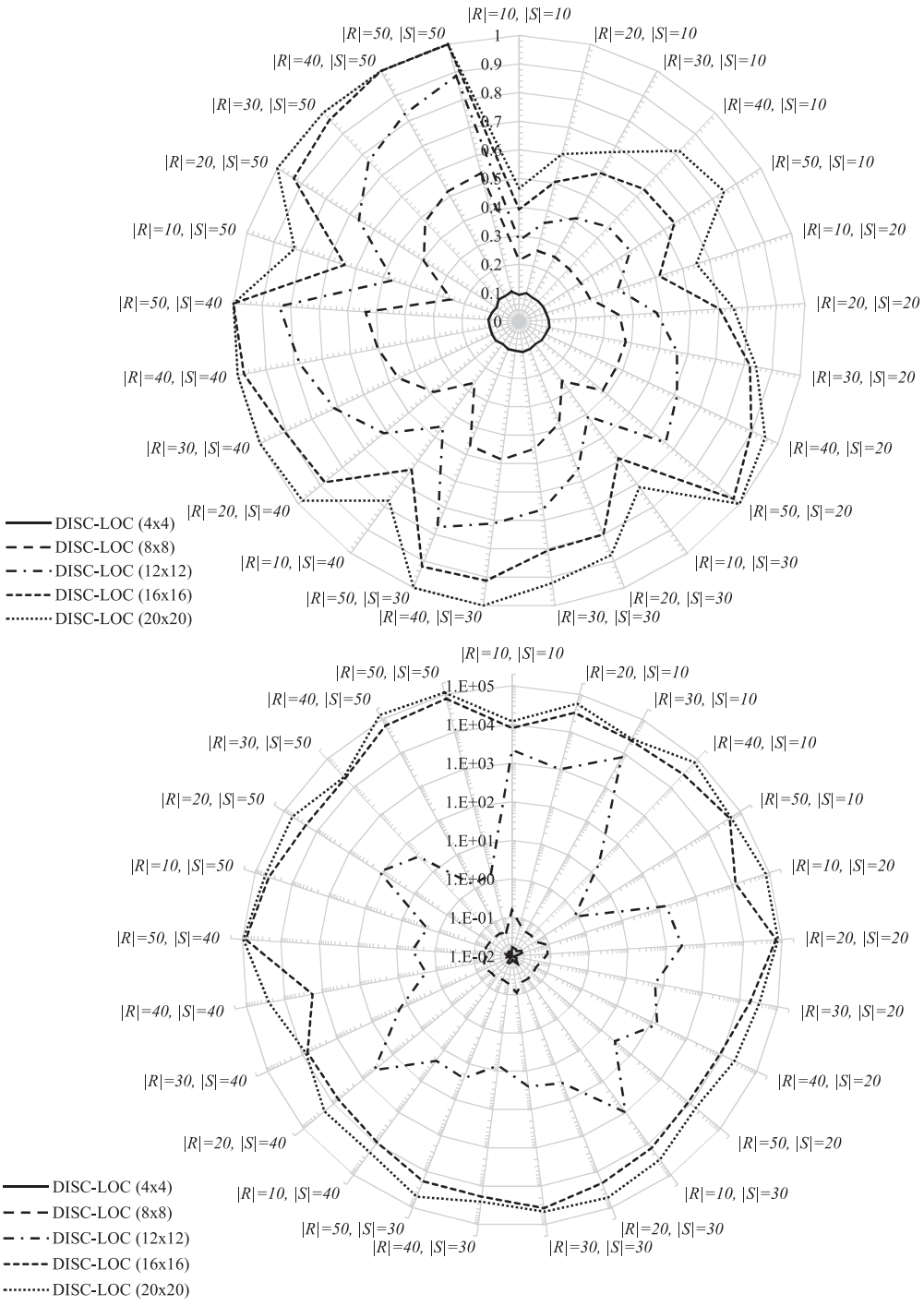


Fig. 10. Performance on DISC-LOC for various grid resolutions. Although DISC-LOC is capable of achieving excellent detection performance at high resolutions (top, fraction of targets detected), it does so at the cost of rapidly increasing computation times (bottom; computation time is indicated in seconds). Note that the bottom figure uses an exponential scale on the radial axis.

optimal solutions to the sensor placement problem, but only near-optimal solutions. In contrast, OPT-LOC's solutions monotonically improve with the iteration steps. As a consequence, Iter-LOC may require significantly more iterations with Greedy-LOC than with OPT-LOC, depending on the termination criteria. In such a situation, the computation time using Iter-LOC with OPT-LOC may not be significantly larger than that using Iter-LOC with Greedy-LOC. Thus, we henceforth use OPT-LOC as a subroutine when comparing the computation times for different algorithms and configurations.

Fig. 13 shows the computation times for DISC-LOC, Adapt-LOC, Iter-LOC, and the C&K algorithm. (The W&K algorithm's computation time is negligible.) The C&K algorithm is the fastest method, while Adapt-LOC needs approximately ten times longer than the C&K algorithm to place both sources and receivers, and Iter-LOC requires up to ten times longer than Adapt-LOC. For example, the computation time for problems with 200 targets and 50 sources and receivers can take up to 300 seconds to achieve a locally optimal solution compared to approximately 7 seconds with the C&K algorithm. Although DISC-LOC outperforms all other algorithms in terms of detection rate, its computation times are approximately 770 times longer than Iter-LOC's for all configurations considered.

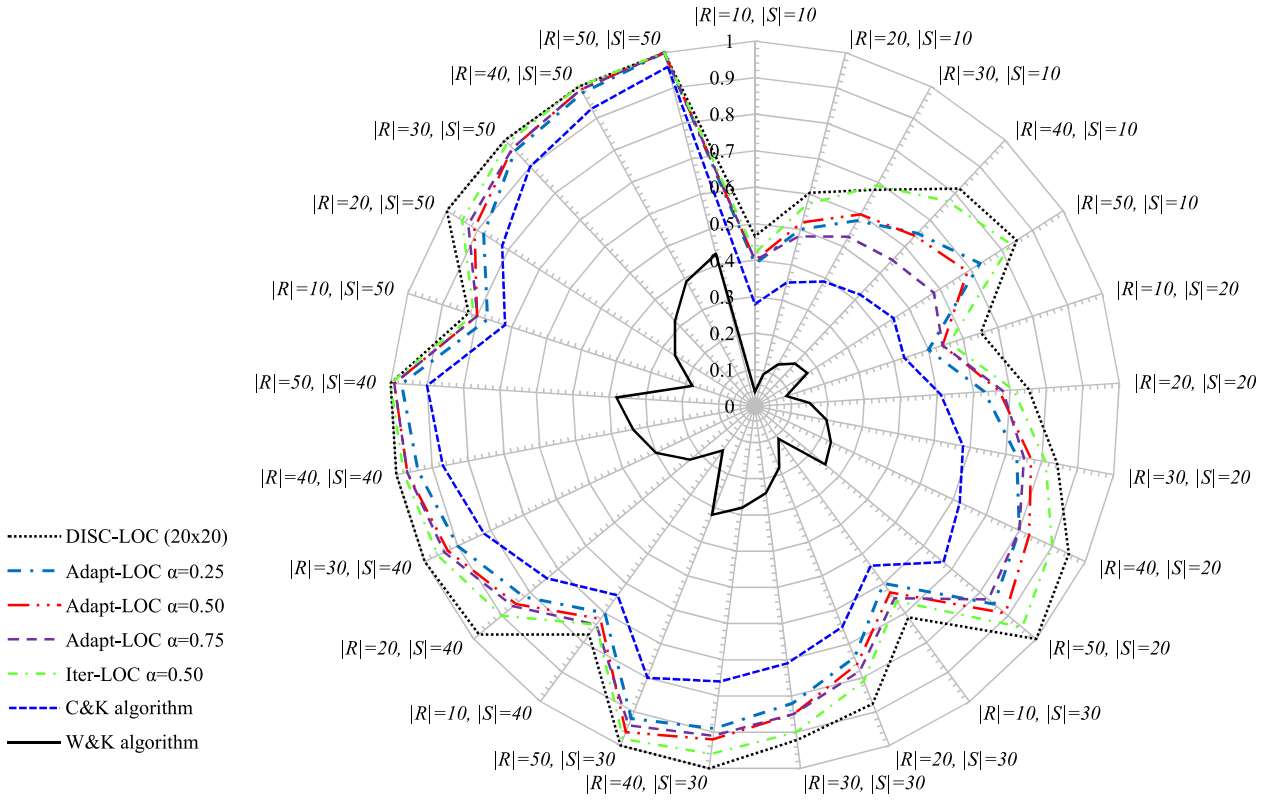


Fig. 11. Fraction of targets detected using DISC-LOC (20×20 resolution), Iter-LOC, Adapt-LOC, the C&K algorithm, and the W&K algorithm.

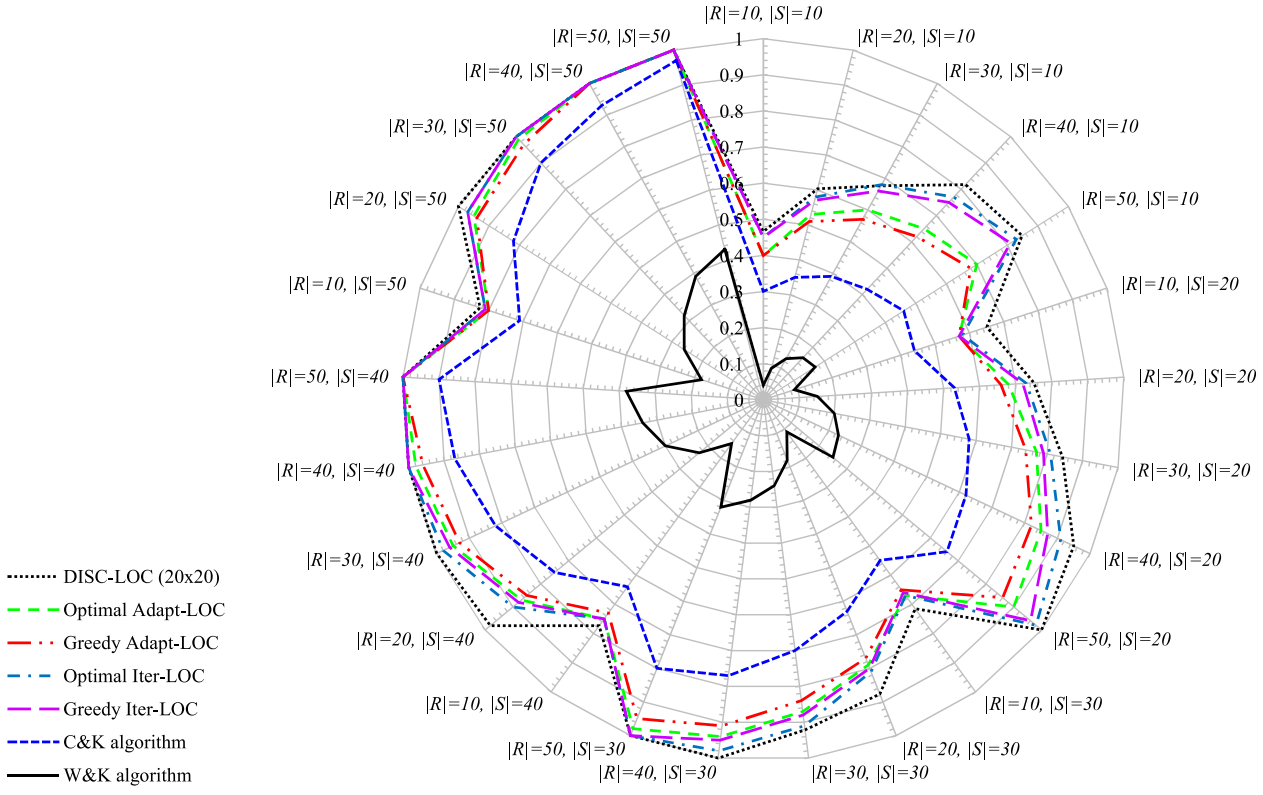


Fig. 12. Impact of the selection subroutine: fraction of targets detected when using OPT-LOC and Greedy-LOC as subroutines for Adapt-LOC and Iter-LOC. The particular subroutine used makes little difference in the extent to which Adapt-LOC and Iter-LOC outperform the C&K algorithm.

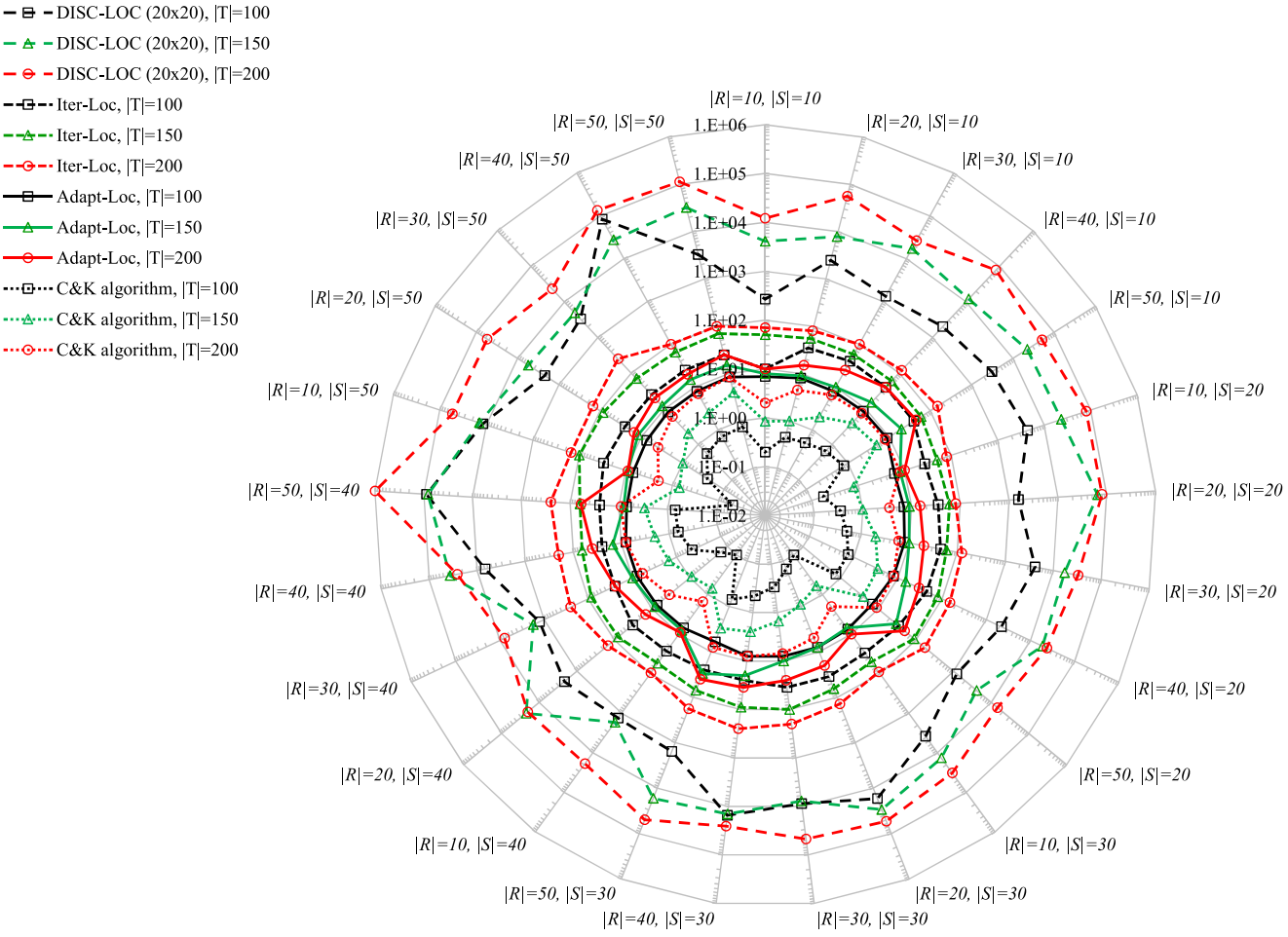


Fig. 13. Mean computation time for $|T| = 100, 150, 200$; $|R|, |S| = 10, 20, \ldots 50$ and $\alpha = 0.5$ for DISC-LOC (20×20 resolution), Iter-LOC Adapt-LOC, and the C&K algorithm.

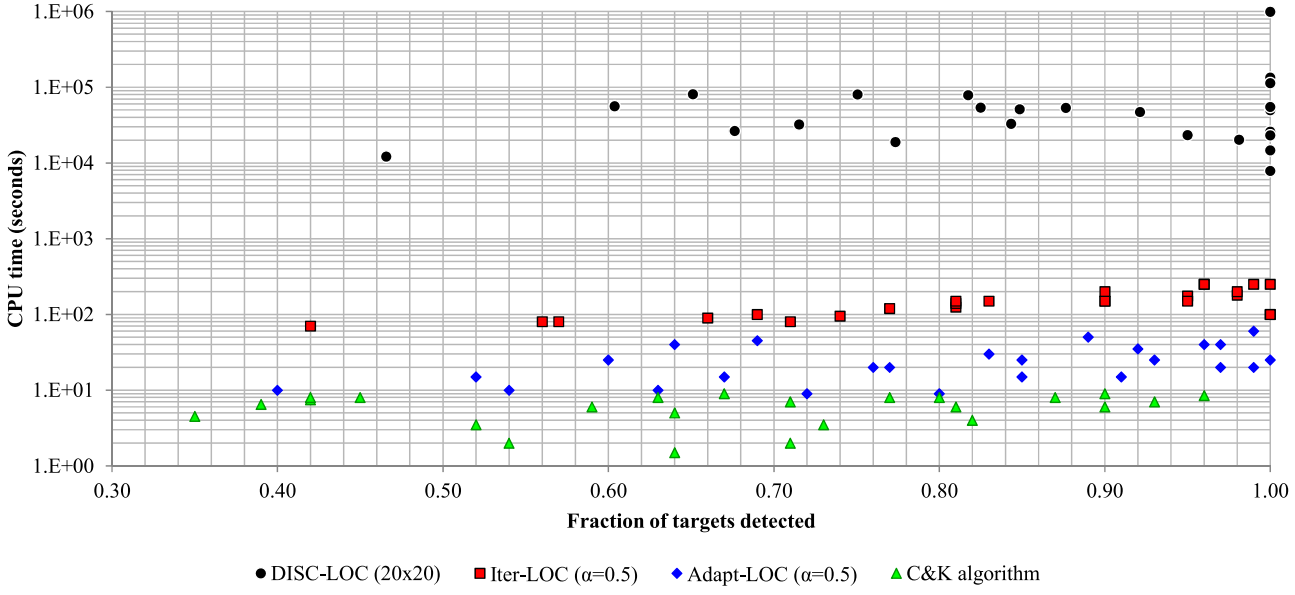


Fig. 14. Performance comparison of DISC-LOC (20×20 resolution), Iter-LOC (α=0.5), Adapt-LOC (α=0.5), and the C&K algorithm in terms of objective function value and computation time. Each marker for a given solution method represents the average performance of that method over 100 randomly generated problem instances for a particular problem size. Although DISC-LOC achieves the best detection performance, it requires by far the most computation time (note that the vertical axis uses a logarithmic scale). In all instances, Iter-LOC and Adapt-LOC use the OPT-LOC subroutine.

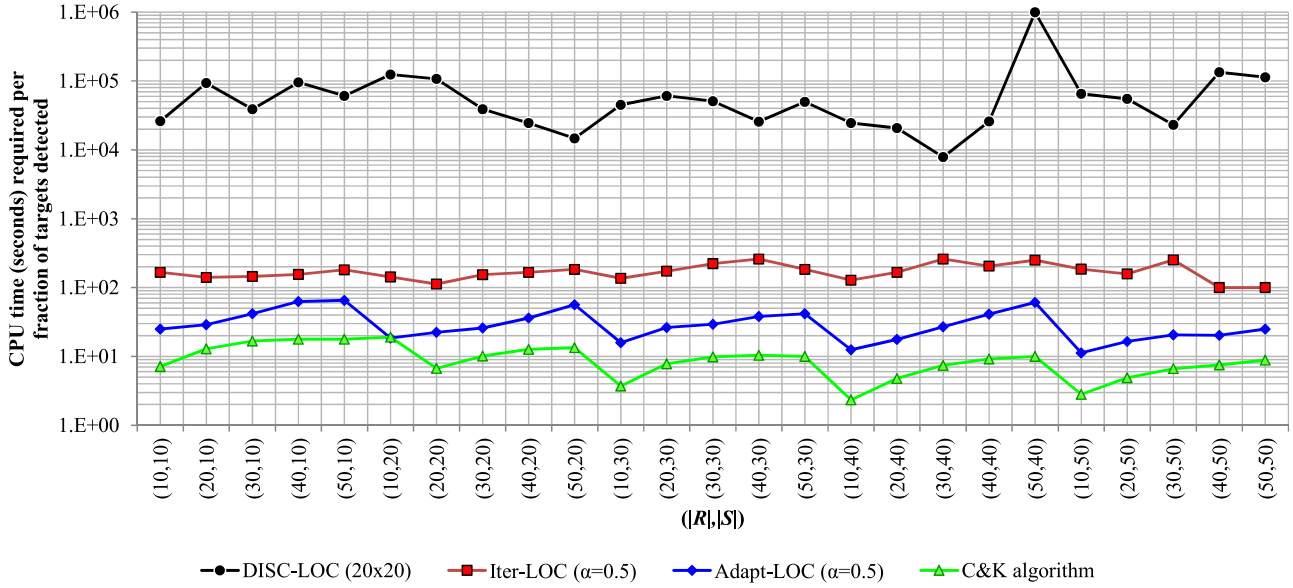


Fig. 15. Performance comparison of DISC-LOC (20×20 resolution), Iter-LOC (α=0.5), Adapt-LOC (α=0.5) and the C&K algorithm in terms of computation time required per unit objective function value. The performance of each solution method in terms of computation time (seconds) required per fraction of targets detected for each problem size (averaged over 100 randomly generated problem instances). In terms of this metric, DISC-LOC is consistently outperformed by Iter-LOC, which is in turn outperformed by Adapt-LOC and the C&K algorithm. Again, the vertical axis uses a logarithmic scale. In all instances, Iter-LOC and Adapt-LOC use the OPT-LOC subroutine.

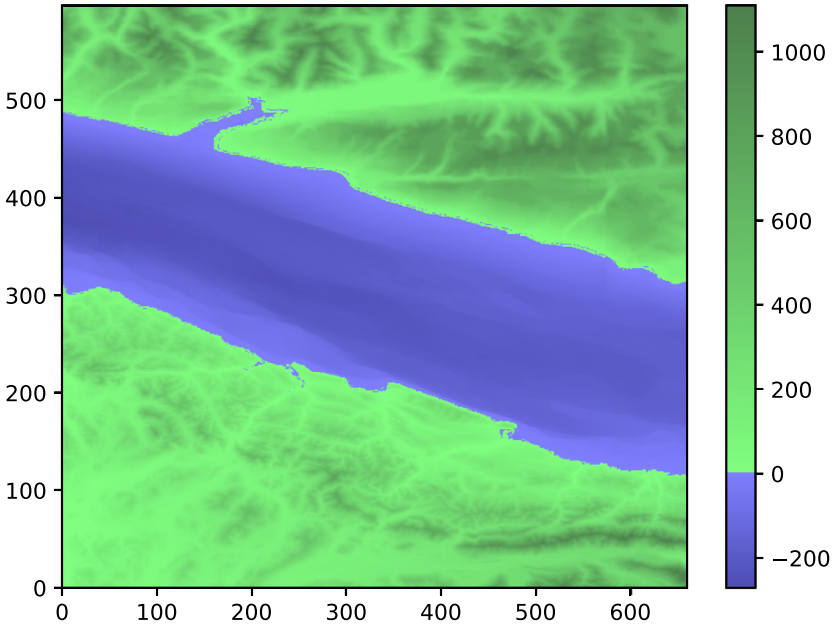
Table 2
Fraction of targets detected and computation times of DISC-LOC (20×20 resolution), Iter-LOC, Adapt-LOC, C&K algorithm and W&K algorithm averaged over 100 randomly generated problem instances for each problem size. The first two columns represent the number of receivers $|R|$ and sources $|S|$ in each instance. The next five columns show the average objective function value achieved by DISC-LOC, Iter-LOC, Adapt-LOC, the C&K algorithm, and the W&K algorithm across all problem instances. Finally, the remaining columns show the average computation times for DISC-LOC, Iter-LOC, Adapt-LOC, and the C&K algorithm. (Because the W&K algorithm places all sensors at random, its CPU time is negligible.) In all instances, Iter-LOC and Adapt-LOC use the OPT-LOC subroutine.

Problem size		Size fraction of targets detected					CPU time (seconds)			
$\|R\|$	$\|S\|$	DISC-LOC (20×20)	Iter-LOC (α =0.5)	Adapt-LOC (α =0.5)	C&K algorithm	W&K algorithm	DISC-LOC (20×20)	Iter-LOC (α =0.5)	Adapt-LOC (α=0.5)	C&K algorithm
10	10	0.47	0.42	0.40	0.28	0.04	12,124	70	10	2
20	10	0.60	0.57	0.52	0.35	0.09	56,224	80	15	4.5
30	10	0.68	0.69	0.60	0.39	0.13	26,407	100	25	6.5
40	10	0.82	0.77	0.64	0.42	0.16	78,463	120	40	7.5
50	10	0.85	0.83	0.69	0.45	0.17	51,378	150	45	8
10	20	0.65	0.56	0.54	0.42	0.09	80,724	80	10	8
20	20	0.75	0.71	0.67	0.52	0.15	80,249	80	15	3.5
30	20	0.84	0.81	0.77	0.59	0.20	32,876	125	20	6
40	20	0.95	0.90	0.83	0.63	0.23	23,331	150	30	8
50	20	1.00	0.95	0.89	0.67	0.25	14,685	175	50	9
10	30	0.72	0.66	0.63	0.54	0.11	32,329	90	10	2
20	30	0.88	0.81	0.76	0.64	0.18	53,447	140	20	5
30	30	0.92	0.90	0.85	0.71	0.24	47,033	200	25	7
40	30	1.00	0.96	0.92	0.77	0.28	25,619	250	35	8
50	30	1.00	0.98	0.96	0.80	0.32	49,851	180	40	8
10	40	0.77	0.74	0.72	0.64	0.15	18,951	95	9	1.5
20	40	0.98	0.90	0.85	0.73	0.23	20,314	150	15	3.5
30	40	1.00	0.96	0.93	0.81	0.30	7846	250	25	6
40	40	1.00	0.98	0.97	0.87	0.34	25,964	200	40	8
50	40	1.00	1.00	0.99	0.90	0.38	993,427	250	60	9
10	50	0.82	0.81	0.80	0.71	0.18	53,796	150	9	2
20	50	1.00	0.95	0.91	0.82	0.26	55,024	150	15	4
30	50	1.00	0.99	0.97	0.90	0.32	23,161	250	20	6
40	50	1.00	1.00	0.99	0.93	0.39	134,047	100	20	7
50	50	1.00	1.00	1.00	0.96	0.43	113,072	100	25	8.5

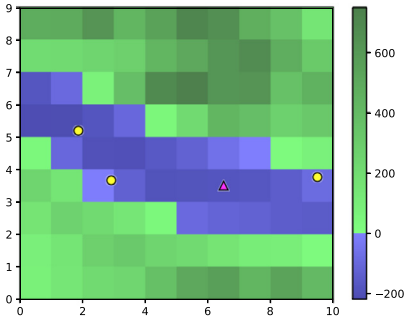
Table 2 summarizes the results of our computational experiments, while Figs. 14 and 15 offer a graphical synopsis of our findings. These figures reinforce our findings that DISC-LOC is capable of providing superior detection performance, but at the cost of computation times that may be prohibitively high in some applications. Adapt-LOC, Iter-LOC, and the C&K algorithm all offer reasonably good performance in a fraction of the computation time, and are often suitable for real-time applications.

4.2. Real-world case studies

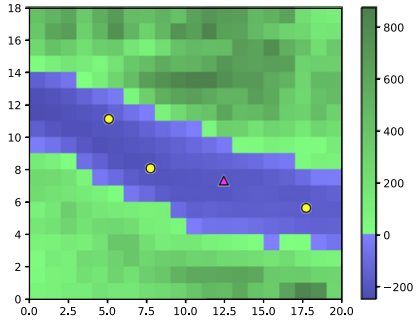
We now compare the performance of DISC-LOC(-enum) and Iter-LOC on a set of real-world examples. We consider three geographical locations: the Strait of Juan de Fuca, which lies between Vancouver Island, Canada, and the Olympic Peninsula in Washington (USA); the ocean in the vicinity of Agadir, Morrocco; and Waccasassa Bay, Florida (USA). For each location, we obtain elevation



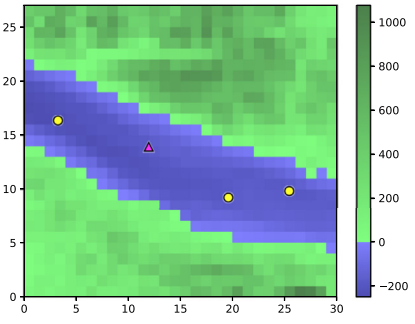
(a) Original data (660 columns, 598 rows).



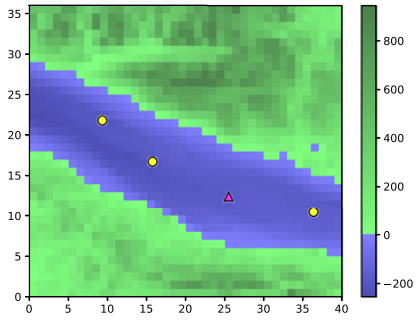
(b) Coarse resolution (10×9 pixels).



(c) Medium resolution (20×18 pixels).



(d) High resolution (30×27 pixels).



(e) Highest resolution (40×36 pixels).

Fig. 16. Test instance "Strait of Juan de Fuca".

Table 3
Performance of DISC-LOC and Iter-LOC on real-world cases. Clearly, Iter-LOC's computation time is substantially lower than DISC-LOC's. Moreover, its detection performance is the same or better than DISC-LOC's for nearly all instances considered. Although DISC-LOC solves to proven optimality, it is restricted to placing sensors on grid points, while Iter-LOC is able to place them in arbitrary locations. This additional flexibility proves to be beneficial in improving the detection performance.

	Problem size				Fraction of targets detected		CPU time (seconds)	
Case name	|R|	|S|	|T|	|D|	DISC-LOC	Iter-LOC	DISC-LOC	Iter-LOC
Strait of Juan de Fuca	3	1	25	3662	0.80	0.84	1	0
	3	1	108	291.043	0.80	0.80	3358	17
	3	1	248	3,457,965	0.79	0.79	15,735	227
	3	I	456	21.259,512	No solution fount	0.76	>6 hours	266
Agadir	3	2	54	18,812	0.74	0.74	9464	4
	3	2	226	1.404,442	0.48	0.68	> 6 hours	116
	3	2	505	15,718,991	No solution found	0.61	> 6 hours	351
Waccasassa Bay	5	2	72	30.953	0.72	0.71	1907	5
	5	2	269	1,893,528	0.43	0.67	> 6 hours	306
	5	2	599	21,352.801	No solution found	0.67	> 6 hours	3002

data from Ryan et al. (2009). The raw data has a very fine resolution, so we aggregate the raw input data into larger rectangular areas (grid cells). Within each cell, we calculate the average elevation data over the pixels in the raw data and apply the resulting elevation to the entire cell. The resulting cells that have negative elevation are under water. The center point of each underwater cell then becomes a target location, as well as a potential sensor location for DISC-LOC, i.e., a member of set G.

For each geographic location we create multiple instances of varying resolutions. Fig. 16 shows an example of the raw data and four problem instances for the Strait of Juan de Fuca. Fig. 16 also shows the solutions obtained by Iter-LOC, where the source is denoted by a triangle and the receivers are denoted by circles.

Table 3 contains the full results of our case study experiments. We set a time limit of 6 hours for DISC-LOC, and we use the greedy subroutine for Iter-LOC. As the table indicates, Iter-LOC's detection performance is comparable to or better than DISC-LOC's for all cases considered, and its computation time is substantially lower. Indeed, Iter-LOC is able to solve all instances at speeds suitable for real-world application. Recall that Iter-LOC is able to place sensors in arbitrary locations, while DISC-LOC is restricted to placing them in predetermined candidate locations. Our results indicate that this additional flexibility allows Iter-LOC to outperform DISC-LOC in terms of coverage, particularly at low resolutions. At higher resolutions we again see Iter-LOC outperforming DISC-LOC because DISC-LOC terminates upon reaching its time limit. As the grid resolution increases, the number of triples $(t, g, g') \in D$ grows so rapidly that the resulting DISC-LOC instances become computationally intractable.

5. Discussion

DISC-LOC, Adapt-LOC, and Iter-LOC are newly developed algorithms that allow intelligent placement of multiple sources and receivers in multistatic sonar environments. They lead to significantly higher target detection results compared to existing algorithms from the literature. Performance gains vary for different mixes of sensors, and the new algorithms outperform the existing algorithms in all experiments with respect to target detection. Although DISC-LOC is capable of the best detection performance, this performance comes at the cost of substantial computation times. Adapt-LOC and Iter-LOC, on the other hand, are capable of producing good solutions much more quickly. The size of the target discs in the first step of Adapt-LOC is crucial for its performance. Excessively large or small discs in the first step can lead to reduced performance in the second step, and it is challenging to determine a universally effective value for α. Fortunately, Iter-LOC eliminates the dependency between the final detection performance and α. Additionally, our experiments indicate that Iter-LOC enables a meaningful gain in target detection in comparison to Adapt-LOC, regardless of the value of α chosen.

Using Greedy-LOC as a subroutine leads to slightly reduced detection rates in comparison to OPT-LOC. However, the fact that Greedy-LOC does not necessarily lead to monotonically increasing detection performance results in a significantly higher number of iteration steps compared to OPT-LOC, depending on the termination criteria. Because of this, it may be simpler and faster to use OPT-LOC than Greedy-LOC.

Compared to the C&K algorithm, Adapt-LOC and Iter-LOC require approximately 10 and 100 times more computation time, respectively, in our experiments. However, they result in approximately 35% and 46% better detection performance, making these two new algorithms the methods of choice when sufficient computational resources are available. DISC-LOC, on the other hand, can provide even better performance at high resolutions. Thus, the user must consider how time-critical the application is, then decide which method is most appropriate.

6. Conclusions

We have described the DISC-LOC, Iter-LOC, and Adapt-LOC algorithms for placing multiple sources and receivers in multistatic sonar environments. Our computational experiments indicate that these algorithms constitute a powerful portfolio of sensor placement techniques, enabling the user to select the most appropriate balance of detection performance and computational speed for a variety of practical situations.

Acknowledgments

The authors thank Prof. Robert Dell for his valuable input, and they thank the anonymous reviewers for their constructive feedback. Dr. Craparo is funded by the Office of Naval Research.

Appendix A

Table 4
Acronyms used in this paper.

Acronym	Meaning	Remarks
Adapt-LOC	Adaptive location	Heuristic for deciding source and receiver locations
C&K	Craparo and Karatas	Determines optimal positions for one type of sensor after deploying the other type at random
CFLP	Continuous facility location problems	
CMRE	Centre for Maritime Research and Experimentation	
DISC-LOC	Discrete location	Determines the best locations for sources and receivers from among a discrete set of candidate locations
FLP	Facility location problem	
Greedy-LOC	Greedy location	Greedily selects positions for one type of sensor, assuming the other type has already been placed
Iter-LOC	Iterative location	Determines locally optimal positions for sources and receivers
LOC-GEN	Location generate	Generates a set of candidate locations for one type of sensor, assuming the other type has already been placed
LOC-GEN-II	Location generate-II	Performs the same function as LOC-GEN, but more efficiently
MSN	Multistatic sonar network	
MSTPA	Multistatic tactical planning aid	
OPT-LOC	Optimal location	Determines optimal positions for one type of sensor, assuming the other type has already been placed
RoD	Range of the day	
UAV	Unmanned air vehicle	
W&K	Washburn and Karatas	Deploys all sensors at random

References

Arabani, A. B., & Farahani, R. Z. (2012). Facility location dynamics: an overview of classifications and applications. *Computers & Industrial Engineering, 62*(1), 408–420.

Bowen, J. I., & Mitnick, R. W. (1999). A multistatic performance prediction methodology. *Johns Hopkins APL Technical Digest, 20*(3), 425.

Brimberg, J., Juel, H., Körner, M.-C., & Schöbel, A. (2015). On models for continuous facility location with partial coverage. *Journal of the Operational Research Society, 66*(1), 33–43.

Carlo, H. J., Aldarondo, F., Saavedra, P. M., & Torres, S. N. (2012). Capacitated continuous facility location problem with unknown number of facilities. *Engineering Management Journal, 24*(3), 24–31.

Casbeer, D. W., Swindlehurst, A. L., & Beard, R. (2006). *Connectivity in a UAV multi--static radar network*: 6209. American Institute of Aeronautics and Austronautics AIAA.

Church, R. L. (1984). The planar maximal covering location problem. *Journal of Regional Science, 24*(2), 185–201.

Cox, H. (1989). Fundamentals of bistatic active sonar. In *Underwater acoustic data processing* (pp. 3–24). Springer.

Craparo, E. M., & Karatas, M. (2018). A method for placing sources in multistatic sonar networks. *Technical Report, NPS-OR-18-001*. Monterey, CA: Naval Postgraduate School. URL https://calhoun.nps.edu/handle/10945/56556694.

Craparo, E. M., Karatas, M., & Kuhn, T. U. (2017). Sensor placement in active multistatic sonar networks. *Naval Research Logistics (NRL), 64*(4), 287–304. doi:10.1002/nav.21746.

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

George, C., & DelBalzo, D. R. (2007). Tactical planning with genetic algorithms for multi-static active sonobuoy systems. In *Proceedings of the nineteenth international congress on acoustics*. Sociedad Espanola de Acustica (SEA).

Gong, X., Zhang, J., Cochran, D., & 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* (pp. 49–58). ACM.

He, Z., Fan, B., Cheng, T., Wang, S., & Tan, C. (2016). A mean-shift algorithm for large-scale planar maximal covering location problems. *European Journal of Operational Research, 250*(1), 65–76.

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

Incze, B. I., & Dasinger, S. B. (2006). A Bayesian method for managing uncertainties relating to distributed multistatic sensor search. In *Proceedings of the ninth international conference on information fusion* (pp. 1–7). IEEE.

Kalkuhl, M., Wiechert, W., Nies, H., & Loffeld, O. (2008). Simulation based optimization of bi- and multistatic SAR missions. In *Proceedings of the seventh european conference on synthetic aperture radar (EUSAR)* (pp. 1–4). VDE.

Karatas, M., & Craparo, E. M. (2015). Evaluating the direct blast effect in multistatic sonar networks using monte carlo simulation. In *Proceedings of the winter simulation conference (WSC)* (pp. 1184–1194). IEEE.

Karatas, M., Gunal, M. M., & Craparo, E. M. (2016). Performance evaluation of mobile multistatic search operations via simulation. In *Proceedings of the forty-ninth annual simulation symposium* (pp. 110–115). Pasadena, CA, USA: Society for Computer Simulation International.

Matisziw, T. C., & Murray, A. T. (2009). Siting a facility in continuous space to maximize coverage of a region. *Socio-Economic Planning Sciences, 43*(2), 131–139.

Ngatchou, P. N., Fox, W. L., El-Sharkawi, M., et al. (2006). Multiobjective multistatic sonar sensor placement. In *Proceedings of the IEEE congress on evolutionary computation (CEC)* (pp. 2713–2719). IEEE.

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

Redondo, J. L., Fernández, J., García, I., & Ortigosa, P. M. (2009). Solving the multiple competitive facilities location and design problem on the plane. *Evolutionary Computation, 17*(1), 21–53.

Revelle, C. S., Eiselt, H. A., & Daskin, M. S. (2008). A bibliography for some fundamental problem categories in discrete location science. *European Journal of Operational Research, 184*(3), 817–848.

Ryan, W., Carbotte, S., Coplan, J., O'Hara, S., Melkonian, A., Arko, R., et al. (2009). Global multi-resolution topography synthesis. *Geochemistry, Geophysics, Geosystems, 10*(3), Q03014. doi:10.1029/2008GC00233.

Strode, C. (2011). Optimising multistatic sensor locations using path planning and game theory. In *Proceedings of the IEEE symposium on computational intelligence for security and defense applications (CISDA)* (pp. 9–16). IEEE.

Tharmarasa, R., Kirubarajan, T., & Lang, T. (2009). Joint path planning and sensor subset selection for multistatic sensor networks. In *Proceedings of the IEEE symposium on computational intelligence for security and defense applications (CISDA)* (pp. 1–8). IEEE.

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

Walsh, M. J., Wettergren, T., et al. (2008). *Search performance prediction for multistatic sensor fields*. IEEE.

Washburn, A., & Karatas, M. (2015). Multistatic search theory. *Military Operations Research Journal, 20*(1), 21–38.

Wong, S., & Sun, S. (2001). A combined distribution and assignment model for continuous facility location problem. *The Annals of Regional Science, 35*(2), 267–281.