Delivery-Aware Directional-Network-Topology Planning for Mobile-Target Search

Shiraz Wasim, Zendai Kashino, Goldie Nejat, and Beno Benhabib

Abstract

In this paper, a novel directional-sensor network deployment strategy is presented and applied to a mobile-target search problem, within a wilderness search and rescue (WiSAR) scenario. This problem has typically been addressed using omnidirectional sensors. Herein, the problem is addressed using directional sensors, utilizing the concept of linear sensing models to simplify network planning and considering physical constraints of sensor delivery to ensure the deliverability of the planned network.

The proposed strategy uses probabilistic target-motion models combined with a variation of a standard direct search algorithm, pattern search, to plan the optimal poses of directional-sensors which maximize the likelihood of target detection.

An illustrative experiment is used to demonstrate the method’s operation. Additionally, extensive statistical simulations were performed to validate the method. A sensing model study justifies the use of a linear sensing model within the planning algorithm. A comparative study highlights the competitiveness of our approach versus three traditional deployment strategies: a uniform, a random, and a ring of fire type deployment.

Keywords – Directional-sensor, sensor delivery, mobile-target search, WiSAR, pattern search

# Introduction

Wireless Sensor Networks (WSNs) have been used to effectively monitor physical phenomena in real-time, where they collect, transmit, and process information in an on-line manner [1]. Their applications include environmental monitoring, [2]–[6], border security [7], target tracking and localization [8]–[10], urban search and rescue (USAR) [11], [12] and lost person detection in wilderness search and rescue (WiSAR) [13], [14]. Research in the field, however, often assumes omni-directional sensing models for network-topography planning, which does not hold true for many sensor types, such as video or infrared, which have a directional sensing range [15].

Sensor networks that utilize directional sensors are, typically, referred to as directional sensor networks (DSNs). In DSNs, a sensor’s sensing model is characterized by a limited angular range in addition to a distal range. Thus, in 2D space, the sensing area of a directional sensor is often modelled as a circular (partial) sector. This contrasts with the sensing area of an omnidirectional sensor, which is a (complete) disk, which is formed when the angular range of a directional sensor is very wide. Another class of sensing model exists at the other extreme, when the angular range is very small. This sensing model, called the linear sensing model, represents sensors which have a sensing area that has been constrained to a line, such as a laser [16]. Being a simpler representation of a sensing region, it is typically (computationally) easier to plan the deployment of sensors with a linear sensing model using a directional model. We postulate that the deployment of a directional sensor can be sufficiently approximated by a linear sensing model during deployment planning, with minimal loss in performance, for the problem considered herein.

Determining poses (*i.e*., positions and orientations – poses) of directional sensors is an important aspect of DSN deployment planning. Deployment planning strategies are predominantly formulated with the goal of maximizing some form of coverage. Two common metrics are total area coverage (with minimum *holes*) (*e.g*., [17]–[22]) and preferential target coverage (*e.g*., [23]–[27]). These objectives are often coupled with secondary objectives such as ensuring connectivity or increasing energy efficiency.

In [18], for example, a tessellation-based approach was used to deploy directional sensors for maximum area coverage. The sensing model is simplified by altering the curved portions of the sector into discretized line segments, forming a hexagonal approximation to the sector. The sensors are then tiled (*i.e*., placed side by side) into a rectangular deployment area containing polygonal obstacles until either all the sensors are deployed, or the area is fully covered. Coverage holes that naturally develop around obstacles, as a result of the tiling method, are removed by placing auxiliary sensors, which overlap previously deployed sensors.

In [26], directional sensors are deployed to maximize coverage of a number of static target positions (point targets) within a region of interest (RoI). The proposed algorithm is a heuristic that deploys sensors in a sequential manner. The approach starts by determining the distances between the target positions. The coordinates of the target position with the highest number of neighbors is chosen as the optimal sensor position. The orientation of the sensor, placed at these coordinates, is determined by performing a 360 rotation and finding the angle at which the greatest number of neighboring positions are covered. Subsequent sensor poses are determined following the same greedy process.

Another important consideration, in deployment planning, is determining the feasibility of the sensor plan. This ensures that the planned network can be realized by the available delivery resources (i.e. UGV’s). There is some existing research that considers sensor delivery during deployment planning [14], [28]–[38]. Typically, these works try to formulate optimal robot trajectories to deliver sensors to a pre-determined optimal sensor network plan [28], [30]–[33], [37]–[39]. Other research focuses on the converse problem where sensor deployment planning is constrained to the trajectories of the delivery agents [14]. In this work, we incorporate sensor delivery considerations into our planning algorithm.

The abovementioned network-deployment research often considers static objectives and regions that are bounded and limited in size which allows for *substantial* coverage. In this paper, we consider the problem of mobile target detection, within a WiSAR context. The objective is to detect a moving target, in a large and growing RoI, as quickly as possible. The expanding nature of the search means one would have an insufficient number of sensors to cover an appreciable fraction of the search area (*i.e*., the sensor network only provides *sparse coverage*). Furthermore, we consider the case in which sensors are static and cannot be repositioned (relocated or re-oriented) after deployment. This contrasts with many existing coverage-enhancing works which concentrate on the optimization of DSNs after they have been deployed randomly [40], [41]. Namely, they assume that the sensors can be relocated after the initial deployment. In unstructured environments like the wilderness, it is often the case that sensors are static and cannot be repositioned (relocated or re-oriented) after deployment.

In other pertinent research works, for example, on target tracking [16], [42]–[44], the objective is to obtain an accurate estimate of the target trajectory using positional updates provided by sensor measurements. Solutions to this problem, however, are not directly applicable to target detection since they assume that the target’s initial position is an *a priori* known, which is not the case during target detection. In barrier coverage [45]–[48], the problems addressed are a subset of the preferential coverage problem, where sensors are deployed to create a barrier between two RoIs. However, these, typically, consider sensor deployment within a bounded RoI. This paper, thus, considers the problem of deployment planning of sparse DSNs that comprise only of static sensors, with the goal being maximizing the probability of detecting a mobile-target in a time-expanding RoI, such as in WiSAR. While our own and others’ earlier works have addressed the mobile-target detection problem using dynamically deployed sensor networks, these have only been for omnidirectional sensors, [49]–[53].

In the proposed methodology, sensor deployment poses and times are determined to, in order of priority, maximize the number of possible target trajectories the sensors would cover, be feasible for a set of delivery resources, while being adaptable if new information is found. Our work is novel in that it: (1) uses a linear sensing model to plan a directional sensor network (2) addresses the novel maximum intersection problem to maximize the probability of target detection (3) incorporates delivery time constraints directly into the sensor pose optimization to guarantee sensor network deliverability.

# Problem Definition

The problem addressed in this paper is the wilderness search problem of localizing a mobile, un-trackable target (*i.e*., a lost person) in an unbounded and growing region. Our focus is on directional (static) sensor networks, where ‘static’ refers to both the (self) immobility of the sensors as well as our inability to relocate them once deployed. A further constraint is that these static sensors have to be deployed by a set of delivery robots that move at a constant speed.

## Wilderness Search and Rescue

WiSAR is often time-critical, where available search resources would be insufficient to provide a complete (hole-less) distributed coverage; especially, considering that the search area grows with time. As a result, the effectiveness of individual deployed sensors diminishes with increasing search time. This highlights the complexity of the problem and requires a deployment solution that allocates search resources effectively and quickly, in order to maximize the likelihood of target detection.

A wilderness search, typically, begins with the notification of a missing person (the target); subsequently, information regarding their last known position (LKP), time at which they were present at this position, and information regarding the target’s demographics is obtained. The target’s travel since leaving the LKP remains unknown, thus, a probabilistic model of his/her motion through the RoI is required. This model could be derived from the statistical behavior of the overall demographic group to which the target belongs [54]. Given an estimate of possible motions that the target could follow, an effective sensor deployment for detecting the target can, then, be planned.

This work assumes that there exists a parallel search effort for the target by mobile operatives. It is anticipated that these operatives would provide feedback to the central search controller in real-time for a potential re-planning of the network deployment (*e.g*., discovering a clue left by the target). Furthermore, we assume sensors with a high fault tolerance, which have enough power to remain operational for the full duration of the search.

## Line Approximation of a Directional-Sensor

The approximation to a linear sensing model can be determined using the mathematical parameters describing a directional sensor. Geometrically, it can be modelled by a 4-tuple (, , ) [15]: is the position of the node,is its working orientation, is its sensing range, and is the angular sensing range, Figure 1a. Our simplified linear sensing model with the same position, range and orientation, as the directional sensor, but with an angular range set to , is shown in Figure 1b.

|  |  |
| --- | --- |
|  | |
|  |  |
| (*a*) | (*b*) |

Figure 1: (*a*) A traditional directional sensing model; and, (*b*) the proposed sensing model (.

This simplified model is only used to reduce the computational complexity of our planning algorithm. The final solution for the optimal positions and orientations of the linear model is used to construct the directional sensors. The network performance is then evaluated, in Section IV, with these directional sensors. Herein, the directional sensors that are constructed from a linear sensor plan are referred to as linear-directional sensors.

## Optimization Objective Function

The objective of DSN planning is to determine sensor deployments that maximize the probability of detecting the mobile target. Herein, a sensor network deployment plan, , for a set of*n* sensors, comprises the sensor positions, orientations,and corresponding placement times, .The optimization problem at hand can, then, be formulated as:

|  |  |
| --- | --- |
|  | (1) |
|  |  |

where *S* is a binary random variable denoting the success/failure of a search mission (*i.e*., a successful target detection).

One may note that it is, typically, not feasible to determine the search success probability exactly. Thus, herein, an estimator of the success probability is used as the objective function. It is defined as the proportion of equally feasible target motions that would be detected by a search:

|  |  |
| --- | --- |
|  | (2) |
|  |  |

where *nt* is the number of equally likely potential target trajectories, and *ntd* is the number of targets, on their respective trajectories, that would be detected by a sensor in the deployed DSN. Assuming a linear sensing model, Fig. 1b, a moving target would be detected when its trajectory crosses a sensor’s sensing region/range. The proposed method, thus, seeks to maximize the number of equally likely target trajectories the search plan would detect by planning sensor poses at locations with high densities of simulated target trajectories. By doing so, the planned network will maximize the likelihood of a successful target detection as defined by the abovementioned estimator. A mathematical formulation relating the above estimator to sensor pose determination is given in Appendix A.

# Proposed Deployment Methodology

The proposed deployment methodology is a three-phase adaptive approach. First, the strategy determines the (original) deployment poses and times for the sensors at hand, prior to the start of the search (*i.e*., ), in an off-line manner. Next, the original deployment plan is initiated. If during this phase, new information would become available (*e.g*., a clue is found), then, the third phase of the proposed strategy is invoked, where the deployment poses and times for the remaining (undeployed) sensors are re-optimized.

## Phase 1: Original Network Planning

The off-line planning phase starts by generating a set of possible target trajectories starting from the LKP, Step 1.1. The deployment pose of a sensor is optimized to intersect the maximum number of target trajectories: (*i*) first, a deployment region optimization is conducted. Here, an optimal sub-region of the search area, denoting a set of potential deployment locations, is found via approximate sensor deployment location optimization, Step 1.2. Deployment time is incorporated into the search for the sub-region to ensure that it is feasible for a sensor to be delivered there. Then, (*ii*) a sensor pose optimization, within the determined sub-region, is conducted to determine the optimal planned pose, Step 1.3. The above outlined process is repeated until all sensor deployment poses and times are determined, Figure 2.

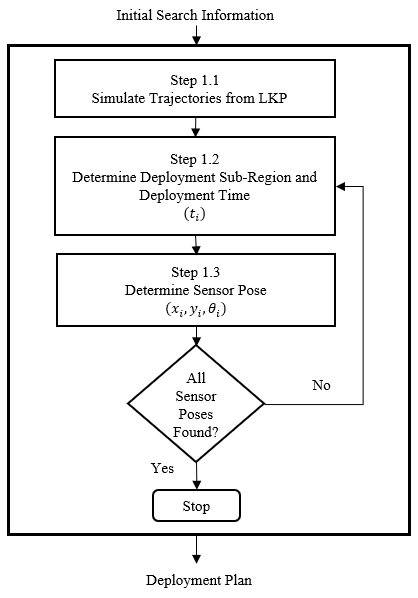


Figure 2: Original network deployment procedure.

The set of trajectories that are intersected, in between each pose determination, are removed and resampled to ensure the same level of information is used to plan each pose. This approach was inspired by the resampling process of a particle filter [55]. A description of the method used to perform the resampling process is described in Appendix B. The above outlined steps are described in more detail in the following sections.

### Step 1.1: Target Trajectory Generation

There exists extensive literature into developing complex probabilistic motion models to predict the behavior of a lost person based on their demographics and environmental features [56]. However, for simplicity, in this work we model the target’s possible motion patterns as randomly generated piecewise linear target trajectories that begin from the target’s LKP [50]. A more detailed description of the target motion model is given in Appendix C. Figure 3a below shows a set of target trajectories that start at the LKP (0, 0) at *t* = 0, and move outward randomly until an end of search time of .

|  |  |
| --- | --- |
|  |  |
| a) | b) |

Figure 3: A set of 100 target trajectories between a) and b) and

Since deployment execution cannot start at the exact moment of notification of a lost target, a specific deployment start time, , from which sensors can start to be deployed is determined by the search commander. Furthermore, as the search cannot continue indefinitely, an end of search time is also defined. As a result, only the portions of the trajectories that occur between and are considered to inform the deployment planning. Figure 3b shows a set of trajectories with a time of 7200s and with the portions of the trajectory before the time of 3600s removed. Thus, the search time duration is defined *a priori* as [, ].

### Step 1.2: Deployment Sub-region and Time Determination

An estimate of the number of trajectories that can be intersected in any region can be determined by examining the density of trajectories in that region. The higher the density, the more likely that a sensor deployed in the region could detect the target. Herein, a hierarchical multi-resolution variation of pattern-search optimization is proposed to determine a sub-region with the highest density of trajectories [57]. A similar hierarchical direct search approach, but using a genetic algorithm, can be seen in [58]. Our strategy uses a hexagonal pattern, shown as the seven black ×s in Figure 4. The objective function evaluated at each point on the pattern is the number of trajectories that pass within a circle centered at the crosses. The region of interest corresponding to each point is shown as a circle, whose size is dependent on the pattern size such that a hole-less coverage is achieved. A more detailed description of the pattern search algorithm, as well as earlier iterations of the pattern search are described in Appendix D. The proposed pattern, in its initial, final and two intermediary configurations, is shown in Figure 4, overlaid above a set of 100 target trajectories.

|  |  |
| --- | --- |
|  |  |
| a) | b) |
|  |  |
| c) | d) |

Figure 4: The hexagonal pattern used to determine the region with the maximum density of trajectories in its a) initial, b) first intermediary, c) second intermediary and d) final configurations

Deployment time is incorporated into the pattern search optimization. For each sub-region in the hexagonal pattern, a corresponding deployment time is determined by calculating the earliest time at which a robot could deliver there, based on each robot’s relative distance and previous delivery assignments. Namely, the deployment time of a sensor, at a given location, is calculated by adding the time it would take for a robot to get to the deployment position to the time at which the robot was at its previous deployment position. Figure 5 shows the positions and times, of three delivery robots, at their previous deployment positions as well as the next potential deployment position (black ×). The distances to the potential position are , , for robot’s , and respectively. For a robot speed of , the earliest deployment times are , 3879s and 3799s respectively. Thus, in this example, the earliest deployment time which is achieved by robot at , would be the deployment time of the sensor. An alternative method of determining deployment time which was investigated earlier in the research and served as a precursor to the one described above is presented in Appendix E.

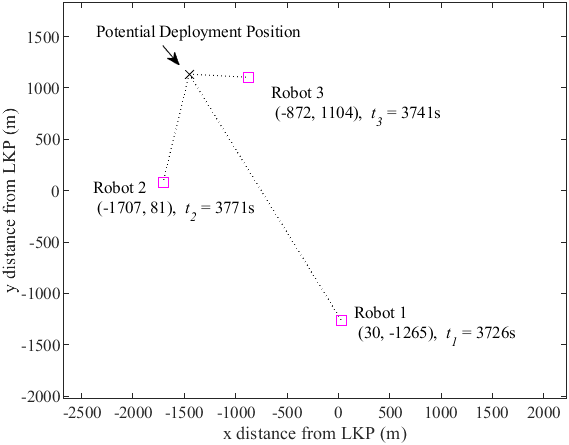


Figure 5: A diagram showing how deployment time is determined

Once a deployment time is determined, only parts of each simulated target trajectory that occur after the sensor becomes active (i.e. when it is deployed) are considered. The optimal deployment sub-region, at each iteration, is the one that contains the highest number of trajectories. If, however, there are no trajectories present within this sub-region, after the determined deployment time, then a deployment in that sub-region is considered infeasible. Namely, the robot would not be able to deliver in time to intercept any of the trajectories passing through that sub-region. It should be noted that this typically only occurs in the initial iterations of the pattern search, due to the large distances of the outer sub-regions from the LKP. In this case, the pattern search algorithm is reinitialized at a position closer to the robots. The process is repeated until a feasible sub-region and corresponding deployment time is found.

### Step 1.3: Sensor Pose Determination

In order to determine the optimal sensor pose, within the optimal sub-region, sensors are positioned such that their sensing line bisects the circular sub-region. Namely, sensors are positioned on the circumference of the circular sub-region determined in Step 1.2 and oriented to point towards the center of the region. An example of this is shown in Figure 6 for three sensor orientations spaced by angles of .

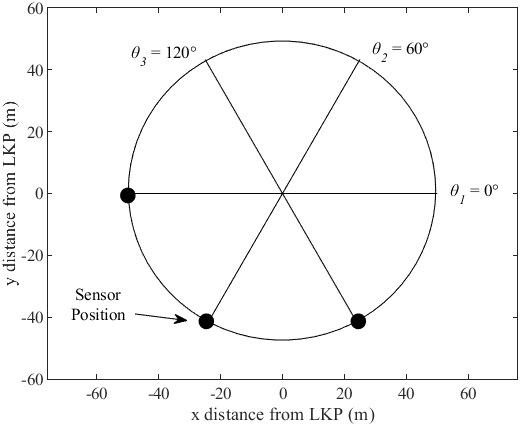


Figure 6: A set of candidate orientations that are evaluated in the optimization process.

The optimal orientation, i.e. the one that intersects the most number of trajectories, is found using a simple search engine [59]. A description of how the number of intersections is determined is shown in Appendix F. As the number of target trajectories intersected by a sensor is a discrete function, the optimization is initialized at several different orientations to avoid converging to a local maximum. A comparison to alternative search engine’s is presented in Appendix G.

After optimal sensor poses are determined, the planned sensor poses are converted from a linear sensing model to a directional model. Namely, the sensor position, orientation and sensing range combined with the directional sensing angle, , are used to form the DSN. An example of the conversion from a (a) linear to a (b) linearly planned directional (linear-directional) sensor network for a set of sensors with a sensing angle of , is shown in Figure 7.

|  |  |
| --- | --- |
|  |  |
| a) | b) |

Figure 7: A sensor network with a a) linear sensing model and b) a linear-directional sensing model

### The linear-directional sensors shown in Figure 7 above are subsequently used to test the performance of the networks. Examples of this are shown in the following sections.

### Deployment Example

An example DSN deployment plan obtained via our proposed strategy for sensors each with a sensing range is given in Table 1 and shown in Figure 8. Optimal sensor poses and times were determined for and The trajectories followed by the delivery robots to deploy this sensor network are shown in Appendix H.

Table 1: The Sensor-Deployment Poses and Times for the DSN shown in Fig. 6.

|  |  |  |  |
| --- | --- | --- | --- |
| Sensor # | Time of Deployment (s) | Deployment Position (m) | Sensor Orientation (rad) |
| 1 | 3683 | (896, 1418) | 1.55 |
| 2 | 3700 | (1794, -912) | 0.82 |
| 3 | 3701 | (355, 1994) | 1.16 |
| ⋮ | ⋮ | ⋮ | ⋮ |
| 28 | 4287 | (-2011, -847) | 2.18 |
| 29 | 4293 | (-2189, 197) | 5.89 |
| ⋮ | ⋮ | ⋮ | ⋮ |
| 48 | 4757 | (104, 2475) | 1.98 |
| 49 | 4807 | (-887, 2342) | 3.84 |
| 50 | 4902 | (-2524, 604) | 6.91 |

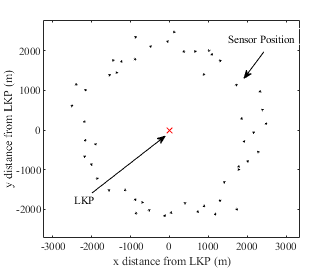


Figure 8: An original DSN deployment plan.

## Phase 2: Network Deployment Execution

This phase of the proposed strategy involves executing the DSN deployment plan determined in Phase 1. Namely, the sensors at hand are deployed according to the optimal schedule at their optimal poses by the assigned delivery robots. If, however, during the original deployment, new information was to become available regarding the target’s motion, for example, through a clue find, then, Phase 3 of the proposed deployment strategy would be invoked, followed by a deployment execution of the remaining sensors at their newly planned poses and times.

## Phase 3: Network Re-Planning

In this phase, the deployment of the remaining (undeployed) sensors is re-planned around the new LKP. Re-planning is important as making use of the updated knowledge regarding the target location could result in a significantly higher probability of target detection. Re-planning determines new deployment poses and times for the set of undeployed sensors, while considering the past deployment.

This phase follows the same stages as Phase 2. Figure 9 shows an example of an original deployment plan (*i.e*., black sensors) and the re-planned network (*i.e*., magenta sensors) as a result of a clue find, at the blue cross, using sensors with a sensing range.

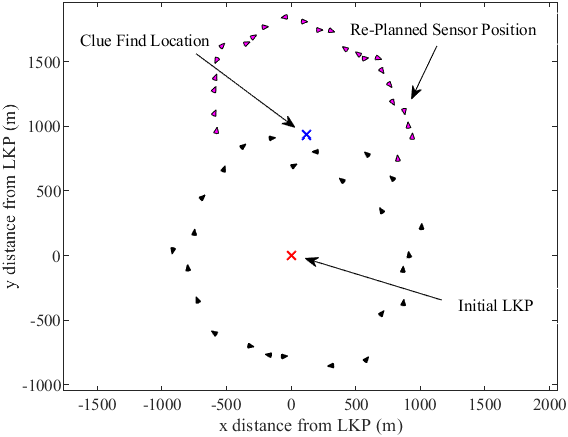


Figure 9: DSN redeployment after a clue find.

# Simulations

Extensive simulated search experiments were performed to demonstrate the effectiveness of the proposed deployment methodology. The subsections below first present an illustrative simulated search experiment to demonstrate the method’s operation. Then, the results of a sensing model case study validating the use of the constrained linear sensing model for deployment planning are presented. A delivery robot performance study highlights the relationship between the quality of delivery resources and the effectiveness of the planned sensor network. Finally, a comparative study contrasting the performance of a sensor network planned by the proposed strategy against the performance of alternative sensor network topologies is presented to highlight the effectiveness of the proposed deployment methodology.

## Illustrative Search Example

An illustrative virtual search experiment, described below, demonstrates the progression of an actual search and how search resources would be deployed. The search begins with the notification of a lost person, last known to be at , at a time . The parameters used to simulate the lost person’s trajectory were a speed of , in a general direction of . The target wanders from the general direction by rad and the maximum distance the target will walk along the chosen particular heading is . The entirety of the path followed by the target is shown in Figure 10. As can be seen, the terrain also contains large-scale obstacles, such as a lake or a dense forest, denoted by the black outlined shapes in the figure.

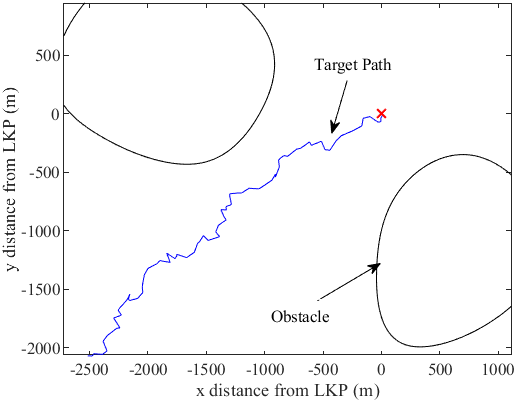


Figure 10: The target path if not intercepted by the search

### Phase 1: Offline Network Planning

The offline network is planned using sensors, each with a sensing range of The search resources are available to deploy at and the search commander has determined an end of search time of .

The network-deployment planning begins with the simulation of 10,000 target motion trajectories, in Step 1.1, using the parameters of the demographic group to which the lost person belongs. For our example, the target walking speeds along the potential paths were sampled from a normal distribution with a mean of and a standard deviation of . The segments, of the target motions, were created using and . These parameters are the same as that of our target to ensure the simulated trajectories reflect the characteristics of their demographic.

Sensor poses were planned sequentially by first determining an optimal sub-region and corresponding deployment time, in Step 1.2, then determining an optimal pose within this sub-region in Step 1.3 and finally constructing the linear-directional sensor from the linear one planned in the previous step. This yielded the positions orientations, and corresponding placement times, for all sensors in sorted order with being the earliest deployment and being the latest. The original deployment plan, around the LKP, is shown in Figure 11.

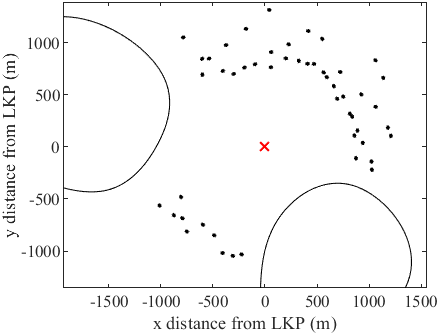


Figure 11: The original deployment plan

### Phase 2: Network Deployment Execution

The deployment plan is then executed, deploying each sensor sequentially starting with the earliest deployment time . However, a clue is found at at . Figure 12 shows the network at various deployment times and the progression of the target trajectory between these times up until the clue is found, denoted by the blue cross. At this time, of the original sensors have been deployed. As a result of the clue find, the third phase of the deployment is invoked where the network is re-planned for the remaining sensors according to the updated LKP.

|  |  |
| --- | --- |
|  |  |
| a) | b) |
|  |  |
| c) | d) |

Figure 12: The search at various times: a) 1876s b) 1946s c) 2130s and d) 2375s

### Phase 3: Network Re-Planning

A conservative estimate of the expected time of arrival, using the maximum target speed and the straight-line distance to the clue position, of the target is . A deployment plan, centered at this new LKP, is simulated to determine the poses for the remaining sensors. Figure 13 shows the re-planned poses of the sensors.

The deployment plan is restarted at . The search continued until the target was eventually detected by a re-planned sensor at , at the position Figure 14 shows a four-frame movie strip from until the time of interception .

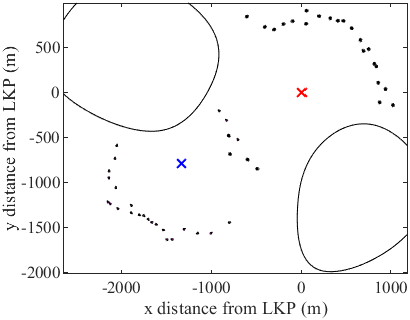


Figure 13: The full deployment including the original and re-planned sensors

|  |  |
| --- | --- |
|  |  |
| a) | b) |
|  |  |
| c) | d) |

Figure 14: The search at various times after the clue find: a) 2448s b) 2512s c) 2686s and d) 4850s

## Sensing Model Case Study

This section compares the use of a linear-directional sensing model to the use of a directional model for search planning (i.e., Phase 1 of the proposed methodology). Specifically, it compares the performance of sensor network topologies planned assuming a linear sensing model during Step 1.3 to one assuming a directional sensing model during the same step. In both cases, the performance of the planned sensor network was evaluated assuming a directional sensing model. To reduce model complexity, the curved portion of the directional sensing model is discretized into straight-line segments. The discretization procedure is described in more detail in Appendix I. This comparison, of sensing models, allows us to determine if incorporating the directional sensing model into our optimization yields better or equivalent results than using a linear sensing model.

The two approaches were tested on a set of trajectories with a deployment start time and end time of and . sensors, with a sensing range of 20m and an angular range of were used for both cases, Figure 15. Visually, the topologies are very similar with sensors spread out fairly evenly around the LKP.

|  |  |
| --- | --- |
|  |  |
| *a)* | *b)* |

Figure 15: Network topologies for the a) linear-directional and b) directional sensing models.

The results for performance by the two models are shown in Figure 16 and Figure 17. As can be seen, the number of targets intercepted by both linear-directional and directional sensors stays fairly even for varying sensing ranges and sensor numbers. Further results, displaying the performance of the two networks at an earlier start time of and a later deployment start time of is shown in Appendix J.

Figure 16: The number of targets found by a set of 50 sensors with varying sensor ranges and a deployment start time of 3600s

Figure 17: The number of targets found by topologies of a varying number of sensors with a sensing range of 20m and a deployment start time of 3600s

Furthermore, a sector angle study was conducted to visualize the change in performance with sensing angle. The number of targets intercepted by the two sensor networks for varying sensing angles is shown in Figure 18.

Figure 18: The number of targets detected by the two sensing models for varying sensing angles

Incorporating the directional sensor into the planning algorithm performs better than the linear-directional sensor at sector angles greater than or equal to . This is because at this angle, the chord drawn between the two outer vertices of the sector, is larger than the radius. Orienting the sensor with this chord perpendicular to the LKP will result in the greatest number of targets intercepted. A visual example of this is shown in Appendix K.

## Delivery Robot Study

The performance of the DSN is dependent on the quality of the available delivery resources. The availability of a larger quantity of robots, or robots with higher speed capabilities, results in a better sensor network plan. This is because a wider range of deployment positions become feasible. Figure 19 displays the number of targets intercepted by DSN’s planned with varying numbers of delivery robots and robot speeds. For each case, the deployments were planned and tested on the same sets of trajectories, using sensors with sensing ranges and a and .

Figure 19: The number of targets intercepted, out of 10,000, by sensor network's planned with a varying number of robots and robot speeds

As expected, in all cases, a larger number of targets is intercepted with delivery robots capable of higher speeds. The number of targets intercepted also increases with a larger quantity of delivery robots. However, at around delivery robots, the number of targets intercepted starts to converge. A possible reason for this is that despite the larger number of robots, the speeds of these robots are not high enough to intercept the time-varying peak of the normal distribution describing the target motion. Thus, in order to improve the performance further, the robot speed would have to be increased.

## Comparative Study against Standard Topologies

To demonstrate its effectiveness, the proposed deployment methodology was evaluated against several existing deployment strategies in a comparative study. However, no static directional sensor deployment methodologies for mobile target detection exist in the present literature. Furthermore, existing DSN literature does not consider sparse coverage with temporal constraints and as a result a direct competitor cannot be established.

Thus, as a benchmark, herein, we compared our proposed strategy against three common deployment methods: a) uniform [60], b) random [61], [62], and c) a ‘ring of fire’ type deployment, which is inspired by the expected topology in a border surveillance application, where search resources are arranged to form a single, air-tight, barrier around a point of interest (i.e., the LKP) [45]–[48].

Uniform sensor networks are typically deployed by dividing the RoI into either square, triangular or hexagonal grid-based patterns and deploying along the grid vertices [60]. In contrast, our approach utilizes a gradient-based optimization [63], using Matlab’s *fmincon* function. The optimization evenly spaces out randomly generated deployment positions within a bounded region until a convergence criterion is met. The size of the bounded region is found by determining the furthest distance a target, with the specified demographics, could have travelled by the end of search time, . Directional sensors are then deployed at these positions and oriented perpendicular to the LKP as this is the orientation expected to maximize trajectory intersections; this approach is also taken in the random and ring of fire deployments.

For the randomly deployed network, sensor positions are found using random distributions. The random distribution for positions has limits to ensure that each sensor is within the specified region; same as that of the uniform deployment. This type of topology is often used within mobile sensor networks as an initial configuration from which adjustments are made to reach an optimal configuration, based on a target metric such as coverage or connectivity [64]. However, as we are only considering static sensors, the random poses of the sensors represent their final configuration.

The ring of fire network is determined by arranging the available sensors into the largest possible, air-tight, circular boundary around the LKP.

Figure 20 shows example topologies of the four deployment methods found using sensors with sensing ranges for a and a . It is assumed that the sensors are deployed all at once for all three comparative strategies.

|  |  |
| --- | --- |
|  |  |
| a) | b) |
|  |  |
| c) | d) |

Figure 20: The topologies of the a) proposed deployment b) uniform deployment c) random deployment and d) ring of fire deployment

All of the comparative models were tested against the same set of trajectories and their performance was evaluated by determining how many trajectories their respective plans was able to intersect (i.e. how likely it was to detect a random target). This was repeated a number of times and averaged to remove performance bias on any particular set of trajectories. The results for the comparative study are shown for a deployment start time of in Figure 21 and Figure 22. The end of search time was .

Figure 21: The number of targets found by a set of 50 sensors with varying sensor ranges with a deployment start time of 3600s

Figure 22: The number of targets found by topologies of a varying number of sensors with a sensing range of 20m and a deployment start time of 3600s

From the results, it is clear that the proposed methodology outperforms all three of the benchmark methods. As expected, a larger sensing range or number of sensors increases the number of targets found. Furthermore, it can be seen that the earlier the deployment starts, a higher total number of targets can be intercepted. This is expected as deploying earlier in the targets’ trajectories means that they will be closer together and a single deployment will be able to intersect a higher number of trajectories. Further results for a deployment planned at start times of and and end times of and respectively are shown in Appendix L.

In the best case, with a deployment start time of and 100 sensors with a 20m sensing range, the proposed methodology outperforms the next best method, the uniform deployment, by 48% (i.e. it intercepts almost twice as many targets). In the worst case, with a start time of and 20 sensors with a 20m sensing range, the proposed strategy is still able to intercept 2.4 times as many targets as the next best model, the uniform deployment.

It should be noted that the ring of fire method will outperform the other methods if a large enough number of resources is available to form an air-tight barrier around all the simulated trajectories. However, the performance of this method diminishes with an increasing deployment start time as the targets have more time to move beyond the barrier. Furthermore, if the target is slower than anticipated, it may be a long time until they reach the ring of fire barrier which is not ideal in a time-critical application such as WiSAR.

In addition to having a higher performance, in terms of target detection probability, the proposed methodology also benefits from being adaptable to new target information. This improves upon traditional static-sensor strategies that deploy at a single optimal deployment time. By being adaptable, the cost of opportunities lost is avoided.

# Conclusions

In this paper, we present a novel static directional sensor deployment strategy for mobile target detection. Optimal deployment poses planned, utilizing the concept of linear sensing models, are determined by positioning sensors in areas of high simulated target densities, in orientations that maximize the number of trajectory intersections. The feasibility of sensor delivery is considered by ensuring that a sensor can be delivered, at the optimal sub-region, by one of the available delivery robots. The proposed strategy is adaptive, in that the deployment plan can be altered if new information becomes available during the network-deployment execution.

A sensing model study validated the approach of using a constrained linear sensor to determine sensor poses. A comparative study with benchmark methods was used to validate our approach. The results showed that the proposed strategy performed the best under the condition of limited resources. Furthermore, it highlighted the importance of an early deployment start time to the performance of the sensor network. A delivery robot performance study is used to show the improved performance by the proposed network when a better quality of delivery resources is available. Further work could include determining if a simplified model could be used to plan the deployment of omnidirectional sensors.

The proposed solution methodology is also applicable to problems beyond WiSAR. For example, the solution to the novel mathematical problem could be used for urban planning where a highway needs to intersect several roads or for planning additional connections to a complex network of pipelines. The density estimation approach, used to determine an optimal sub-region, could be used for applications that require determining priority areas, within a bounded region, to optimize for a specific metric. Conversely, the approach could be used to determine areas to avoid by locating regions with high densities of a specified metric.

# Appendices

## Appendix A: Optimization Metric for Sensor Poses

Simulated target trajectories are used to determine sensor deployment poses to maximize the likelihood of target detection. A function, , can be defined, that describes the number of trajectories intersected by a directional sensor, of a particular sensing range, , if placed in a certain position, at time , and in an orientation . Mathematically, this can be written as:

|  |  |
| --- | --- |
|  | (5) |

where is the scalar value representing the number of trajectories intersected by sensor at a position , at time and in an orientation (). As the probability of success, as defined in equations (1) and (2), is determined by the number of targets detected, maximizing the number of trajectories intersected also maximizes the estimate of the cumulative probability of success. Thus, can be set as the objective function which needs to be maximized. This is described by

|  |  |
| --- | --- |
|  | (6) |

where is the number of trajectories intersected at the optimal position and orientation.

## Appendix B: Trajectory Resampling Process

In this work, the objective is to detect a target traversing along a particular trajectory. As the path the target will take is unknown, it has to be estimated. The proposed method takes a Monte Carlo approach, where a large set of equally likely trajectories are simulated and used to inform sensor deployments. By placing sensors, the trajectories that are intersected can be eliminated from the total set of trajectories that the target can possibly take. As the process only starts with a finite set of possible trajectories, sequentially eliminating them can eventually lead to a depletion of target information. As a result, the trajectories are resampled to maintain a certain level of information for subsequent sensor deployments. The above outlined estimation method, in particular the resampling process, is similar to the approach taken in particle filters and is described in comparison to that in the following.

Particle filtering is a recursive method used to estimate the state of a stochastic process in the presence of noisy or incomplete data [55]. A set of generated particles, or samples, are used as predictions of the state of the system. These predictions are updated, by assigning them an importance weight which represents the probability of that prediction being the correct state. This weight is determined by comparing the predicted state to the actual state using the available secondary information. The particles with the lowest assigned importance weights are removed and re-sampled in proximity of those particles with higher weights. The process is then repeated until a good estimate of the state is achieved. In our approach the particles represent the trajectories in target space. Sensor placements represent updates in the information regarding the target. This information reduces the probability of some of the trajectories (i.e. particles) being the actual trajectory of the target. In contrast to the particle filter method, where the assigned weights can be continuous, a trajectory is assigned a 0 or a 1 weight where a 0 represents a trajectory that is intersected and a 1 represents a trajectory that is not. As a result, the particles with a 0 weight are removed and resampled. By removing trajectories that are intersected in certain locations (i.e. near sensor poses), we are effectively re-sampling the trajectories in locations that contain trajectories of a higher weight, in this case equaling 1, just like in particle filtering. This process is recursive in that trajectories are sequentially eliminated and resampled until all sensor poses are determined.

Practically, this methodology was implemented by first simulating a single set of trajectories. For these, an initial sensor pose and deployment time are determined. The number of trajectories, , that are intersected by this pose are found and removed. Then, the same number of trajectories are re-simulated. A brute force methodology is used where each re-simulated trajectory is tested against the planned sensor pose to ensure they are not intersected. When the required trajectories are found, they are added to the previously non-intersected trajectories. This new set of trajectories are then used to plan subsequent poses. A flow diagram describing this process is shown in Figure 23.

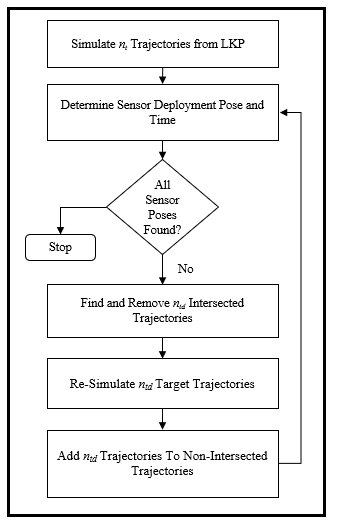


Figure 23: Trajectory re-simulation in deployment planning

An alternative method to obtain re-simulated trajectories, that are not intersected by the already planned sensor poses, involved reducing the range of initial directions that a target could take from the LKP. In the proposed methodology, target trajectories are simulated with an initial heading, , directly away from the LKP. However, when re-simulating trajectories, particular angles of , that were known to be in the direction of planned sensor poses, were removed. These angles were found by finding the angle between the coordinates of the sensor position, , and the LKP. However, as this method did not guarantee that the trajectories produced, using the limited values of , would not intersect any sensors, it was eventually discarded.

## Appendix C: Trajectory Generation Model

The simulated trajectories, used in this work, follow an outward propagation model with a target’s potential proclivity to randomly wander factored in. This is characterized by straight-line segments with probabilistic heading changes. The likelihood of a target walking on a particular heading, , and travelling a distance along that heading, , in its motion, is given by:

|  |  |
| --- | --- |
|  | (3) |
|  | (4) |

where is the maximum distance the target will walk along a heading, is the degree to which the target wanders, and is the heading directly away from the LKP. A normal distribution centered on the direction directly away from the LKP is used to model target heading changes. The outward propagation model is used because historical data suggests that targets continue travelling in their initial direction of motion after leaving the LKP. The velocity of the targets is modelled as a normal distribution centered on the mean target speed and standard deviation

## Appendix D: Pattern Search Description and Alternative Pattern

### Pattern Search Method

A heuristic known as pattern search optimization is used to determine the pose of the sensors. This is a derivative-free direct search algorithm well suited to the discrete trajectory intersection problem being considered in this work. The optimization strategy, in the context of this paper, starts by defining a stencil such as the one shown in Figure 25. Each circle defines a region within which the number of trajectories passing through is evaluated. If the circle, in the center of the stencil, contains the highest density of trajectories, then the size of the stencil is isometrically reduced by a scale factor (how the scale factor is determined is described in the following section). If the maximum value is found on one of the outer circles, then the stencil is shifted so that the circle that contained the highest density of trajectories is now at the center. The density of trajectories, in this new configuration, is then evaluated. This process is repeated until a stopping criterion has been reached. The stopping criterion, for this work, is when the diameter of the circles is the equivalent length of the defined sensing range. After the stopping criterion has been met, the circle with the highest density of trajectories is selected. This circle represents the sub-region that is used to determine the sensor pose.

### Pattern Search Stencil

In an earlier iteration of the pattern design, the geometry was defined as boxes with the center’s forming a cross as shown in Figure 24.

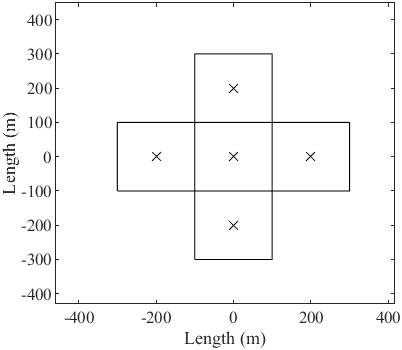


Figure 24: An early version of the pattern search tool

However, once the pattern search algorithm was established, the pattern search tool was altered to reflect the sensing models considered in this paper. Figure 25 shows the hexagonal pattern used in our work.

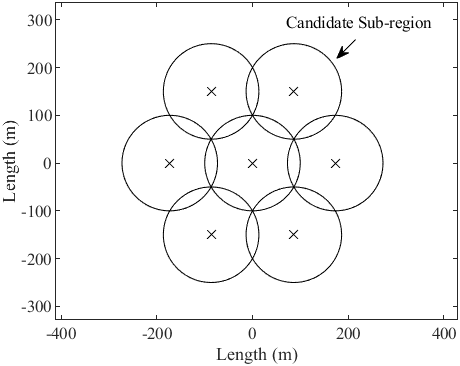


Figure 25: The final pattern used in our proposed methodology

Using a complete disk, with a final diameter that is the same size as the linear sensing range, allows for a more effective orientation optimization as the sensor line can be swept circularly until an optimal is found. This design was inspired by the patterns seen in sensor network deployment problems involving k-coverage, where multiple sensors overlap to ensure every point in the RoI is covered by at least k sensors [65]–[67].

### Pattern Search Resizing

As mentioned, during the search, if the center circle is optimal (i.e. contains the greatest number of trajectories) then the pattern is resized to fit within it, such that a hole-less coverage is achieved. An example of a resized pattern is shown in Figure 26.

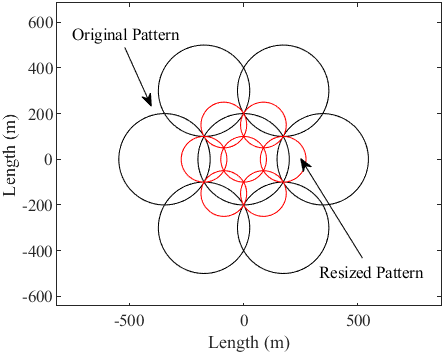


Figure 26: An example of the pattern resizing process

Mathematically, the problem can be defined as determining a value of (i.e. the radius of the circles in the resized pattern) that creates a pattern which fits within the central disk, with radius , without any coverage holes. Figure 27 shows a portion of the resized pattern in red, and the central disk, in black. It is evident that, to achieve a hole-less coverage, the distance, , between the coordinates of the center circle and the intersection point of the two red circles, , must be equal to the radius of the center circle, Only a portion of the pattern needs to be considered due to symmetry.

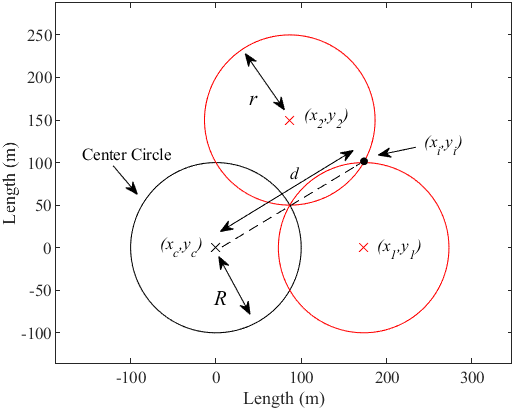


Figure 27: A diagram showing the alternative method to determine deployment time

The intersection point, , can be found mathematically by equating the equations of the two red circles:

|  |  |
| --- | --- |
|  | (7) |
|  | (8) |
|  |  |

where, and are the coordinates of the centers of the two circles. As the pattern is pre-defined, the equations for the center points of these circles are:

|  |  |
| --- | --- |
|  | (9) |

|  |  |
| --- | --- |
|  | (10) |

|  |  |
| --- | --- |
|  | (11) |

|  |  |
| --- | --- |
|  | (12) |

These are described relative to the coordinates of the center circle. The derivation, for the intersection points, yields the following equations:

|  |  |
| --- | --- |
|  | (13) |

|  |  |
| --- | --- |
|  | (14) |

where and are:

The equations for the intersection point, and , can then be substituted into the equation for :

|  |  |
| --- | --- |
|  | (15) |

As this is a complex set of non-linear equations which cannot be easily rearranged for , it was solved with an optimization approach using Matlab’s *fmincon* function. Namely, the value of was initialized and substituted into the above equations to minimize the cost function:

|  |  |
| --- | --- |
|  | (16) |

It was found that the optimal value of is equal to . From this, we can deduce that simply halving the radius of the original pattern size will give the radius for the minimum resized pattern required for a hole-less coverage. However, considering the discrete and random nature of the trajectories, on which the pattern search algorithm is performed, the resizing parameter in our simulations was set to to include the surrounding are in the search.

## Appendix E: Alternative Deployment Time Method

Deployment time, in an earlier method, was found by determining the earliest time a sensor, in a particular pose, was expected to detect a target. This can be determined using the temporal and spatial information available from the simulated target motion models. Namely, as each trajectory is a set of line-segments formed using discrete points, each with an associated time, when an intersection between one of these line-segments and a sensor is determined, the associated time for that particular line-segment is used as the deployment time. This ensured that a sensor was deployed, just in time, and allowed the sensors to be held in reserve for as long as possible in case of a clue find. However, this method did not consider how the sensors would be delivered to their respective positions. It was assumed that a mobile agent was able to deploy them instantly at the required deployment times. To take deliverability into account, the deployment time methodology was expanded to include delivery robots that imposed time constraints on optimal sensor positions, as explained in the proposed methodology.

An example of how a deployment time, in the precursor method, was determined is shown in Figure 28. Herein, three potential target trajectories intersect the deployed sensor’s range line. As can be noted, the earliest target to be detected is the one travelling on Trajectory 3, at , which is chosen as the deployment time for this sensor.

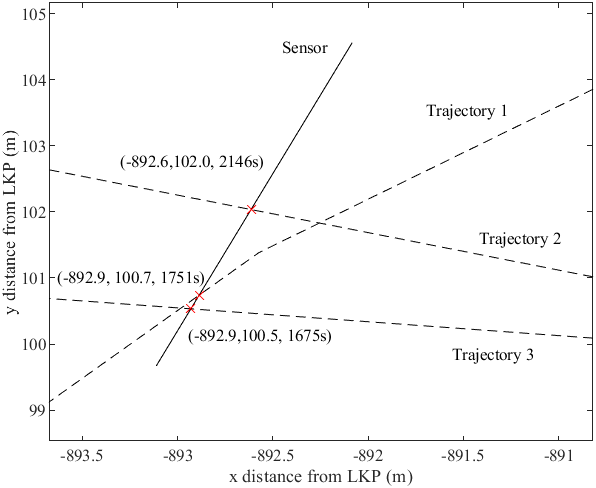


Figure 28: A diagram showing the alternative method to determine deployment time

## Appendix F: Method for Determining Intersections

Determining line intersections is used repeatedly in the proposed methodology and is a core function within the code. As a result, a more detailed description of how it is implemented is given here. As mentioned, the simulated trajectories are modelled as multi-segmented lines. If a trajectory is to be intersected by a sensor, at least one of the multiple segments of a single trajectory must contain an intersection point with the line representing the sensor. This fact can be used to determine how many unique trajectories are intersected by each of the candidate sensor orientations, during orientation optimization, or how many trajectories are intersected during re-simulation or testing. Mathematically, this can be solved using the equations of a pair of line segments [68]. One line is connected between points and and the other is connected between points and . This is described by the following equations:

|  |  |
| --- | --- |
|  | (17) |
|  | (18) |

where is the point of intersection of the two lines and and are the offsets of the intersection point for each corresponding line. In the context of this paper, and represent the end points of a particular segment of a trajectory and and represent the start and end points of the sensor line respectively. By equating the two and solving the system of linear equations, values for and can be obtained. If both values satisfy the inequalities,

then a point of intersection exists between the two-line segments. This point of intersection can be found by substituting the values of and back into either equation’s 1 or 2 respectively.

As performing this calculation on every segment of each trajectory and against every sensor pose was computationally expensive, a reduction strategy was used to only evaluate trajectories that are in proximity of the candidate sensor position. A box defined by upper and lower bounds in both the and directions was generated around the candidate position. A trajectory is considered to pass through this box if at least one set of coordinates pertaining to one of its line-segments is within the defined bounds. In addition, a margin, of the maximum possible step size that a target could take (i.e the maximum possible length of a single line-segment), is added to the bounds to ensure no targets are missed. A snapshot at a particular sensor position with three target trajectories is shown in Figure 29. In this example, the trajectories in solid black pass through the box and are accepted (i.e. there is a possible intersection) and the dashed blue trajectory is rejected (i.e. no possible intersection).

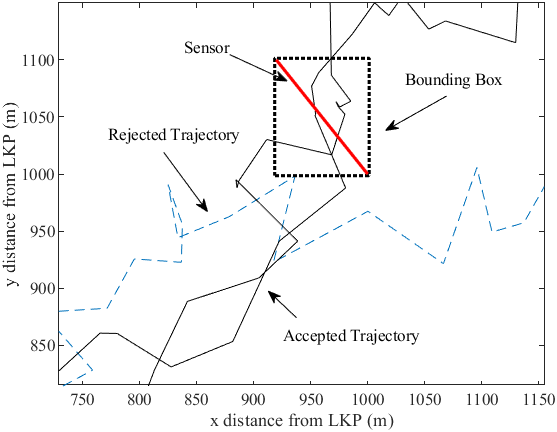


Figure 29: An example of trajectory reduction using the bounding box method

As this method only considers if the discrete coordinates of a trajectory are within the specified bounds, it does not guarantee that the sensor will be intersected. As a result, the mathematical formulation, in equations 17 and 18, is used on the reduced set of trajectories to confirm if an intersection exists.

## Appendix G: Alternative Deployment Orientation Methods

In the proposed methodology, the orientation optimization step is performed using a simple search engine. The search engine used is Matlab’s *fminsearch* function that follows the Nelder-Mead simplex method [59]. The optimal value of is found such that the number of trajectories that are intersected by the sensor is maximized. Other optimization algorithms, available within Matlab’s optimization suite, were all tested and compared to determine which performed the best. These included a brute force method where 64 pre-set orientations were tested and an optimal from within these was chosen, as well as the *fminsearch,* *fminbnd* and *fmincon* functions. As the trajectories are discrete in nature, the algorithm was run multiple times with varying initial values of to avoid convergence to a local minimum. Each of the optimization functions were run with two methods of initialization: random initializations of and equally spaced initializations within the interval .

The performance was tested on 100 sets of target trajectories each with a corresponding sub-region. The number of targets in each sub-region varied so a target detection ratio was defined. The ratio is the number of targets detected out of the total number present in a particular sub-region. This was computed for each sub-region and each optimization approach. The results, for each optimization approach, on all 100 sets was then averaged. In addition, this test was then repeated for various sensing ranges, each of which had its own set of target trajectories and corresponding sub-regions. The results of the study are shown in Table 2.

Table 2: A comparison of the performance of several algorithms on orientation optimization

|  |  |  |  |
| --- | --- | --- | --- |
| Optimization Function | Target Detection Ratio | | |
| Sensing Range | | |
| 20 | 50 | 100 |
| Brute Force (64 Orientations) | 0.730 | 0.644 | 0.653 |
| *fminsearch* (single) | 0.783 | 0.729 | 0.687 |
| *fminbnd* (single) | 0.890 | 0.859 | 0.800 |
| *fmincon* (single) | 0.662 | 0.675 | 0.604 |
| *fminsearch* (random) | 0.940 | 0.904 | 0.849 |
| *fminsearch* (equi-spaced) | 0.950 | 0.923 | 0.849 |
| *fminbnd* (random) | 0.890 | 0.859 | 0.800 |
| *fminbnd* (equi-spaced) | 0.890 | 0.859 | 0.800 |
| *fmincon* (random) | 0.919 | 0.846 | 0.798 |
| *fmincon* (equi-spaced) | 0.903 | 0.869 | 0.797 |

As can be seen, the worst performing algorithms were *fmincon* and the brute force method, achieving the lowest ratios for all three sensing ranges. However, the brute force method would be expected to improve if a larger set of orientations was tested, though with the tradeoff of an increase in computation time. The best performer was *fminsearch* with equi-spaced initializations across all sensing ranges achieving a ratio of in the best-case scenario. As a result, *fminsearch* with equi-spaced initializations was used in the proposed methodology.

It can also be seen that the detection ratio in most cases decreases with sensing range. One possible reason for this is that the size of the sub-region is dependent on the sensing range; a larger sensing range results in a larger sub-region which means a larger quantity of targets that the sensor needs to detect. Furthermore, a larger sub-region could mean that the trajectories are more spaced out and an orientation intersecting a large proportion of them would be more difficult to obtain. As a result, the target detection ratio would decrease.

## Appendix H: Example Robot Trajectories

The deployment example, in Figure 8, is deployed using a set of three robots that move at a constant speed of . The trajectories followed by these robots are shown in Figure 30.

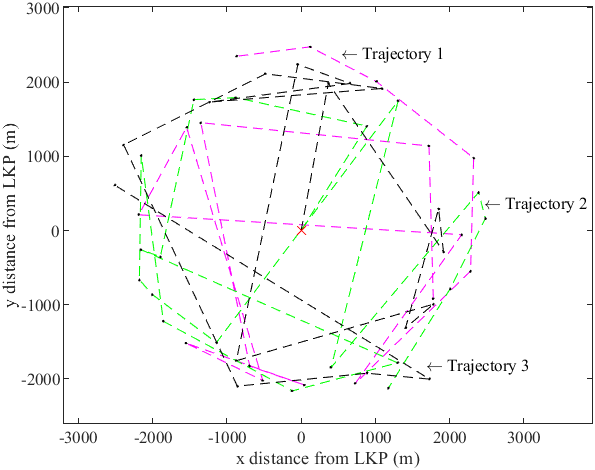


Figure 30: Robot delivery trajectories for three robots

The trajectories do not follow any sort of regular pattern as the deployment positions are determined using a greedy approach, as mentioned in Appendix C.

## Appendix I: Directional Sensor Discretization Study

When constructing the directional sensors using the linear sensor plan, the curved portion of the geometry was modelled as straight-line segments. This reduced the complexity of the model and also enabled the use of the intersection method described in Appendix F. Herein, the straight-line segments used to approximate the curved portion of the sensor are referred to as discretization’s. Two examples, where 1 and 3 discretization’s have been applied to a directional sensor with a sensing angle of and a sensing range of , are shown in Figure 31a and Figure 31b respectively.

|  |  |
| --- | --- |
|  |  |
| *a)* | *b)* |

Figure 31: Directional sensor’s with a) 1 and b) 3 discretization’s

Discretizing the directional sensor reduces the total area covered by it. As a result, a fewer number of targets could potentially be intersected. This was tested by determining the number of trajectories intersected by a single sensor plan with linear-directional sensors constructed with a varying number of discretization’s. In addition, the sensing angle was also varied to study the effect of a low number of discretization’s at large sensing angles. The sensor plan and the target trajectories that were tested on were kept constant between tests. The results are shown in Table 3.

Table 3: The number of targets detected by topologies with sensors of a varying number of discretization’s at varying sensing angles

|  |  |  |  |
| --- | --- | --- | --- |
|  | No. of Targets Detected | | |
| Sensing Angle | | |
| No. of Discretization’s |  |  |  |
| 1 | 833 | 891 | 932 |
| 2 | 837 | 899 | 950 |
| 3 | **838** | **902** | 951 |
| 4 | 838 | 902 | 955 |
| 5 | 838 | 902 | **956** |
| 6 | 838 | 902 | 956 |

As can be seen, for the and sensing angles, only three discretization’s are needed before no improvement in performance, with additional discretization’s, is observed. For the case, at lease five discretization’s are required. As expected, the results show that larger sensing angles require a greater number of discretization’s to avoid loss in performance. However, as this work focuses on homogenous sensors with a sensing range, to reflect the sparsity of resources, the number of discretization’s during the simulations was set to three.

## Appendix J: More Analysis and Graphs of Sensing Model Study

Additional graphs comparing the performance of a planning algorithm utilizing a linear or a directional sensing model can be seen for a start time of in Figure 32 and Figure 33 and a start time of 7200s in Figure 34 and Figure 35. The parameters used to generate these graphs were:

* # of Targets = 10,000
* Mean target speed = 0.7 m/s
* Std. dev. of target speed = 0.23 m/s
* = π/3 rad
* = 100 m
* # of sensors = 20, 50, 100
* Sensor Range: 10, 20, 30m
* Sensing Angle:
* # of Discretization’s: 6
* # of robots = 3
* Robot speed = 10 m/s
* = 1800s, 7200 s
* = 5400s, 10800s

Figure 32: The number of targets found by a set of 50 sensors with varying sensor ranges with a deployment start time of 1800s

Figure 33: The number of targets found by topologies of a varying number of sensors with a sensing range of 20m and a deployment start time of 1800s

Figure 34: The number of targets found by a set of 50 sensors with varying sensor ranges with a deployment start time of 7200s

Figure 35: The number of targets found by topologies of a varying number of sensors with a sensing range of 20m and a deployment start time of 7200s

Further analysis of sensing models compared the performance of a linear sensor network and an omni-directional sensor network (planned using an omni-directional sensing model) with the aforementioned linear-directional and directional topologies. A novelty within this work is the use of a linear sensing model to plan the deployment of a DSN. The analysis in this section explores if a linear sensing model could be used to plan an omni-directional sensor network. For a fair comparison, the diameter of the omni-directional sensors was set equal to the sensing range of the directional sensors. The results are shown in Figure 36 and Figure 37. The parameters used in this study were:

* # of Targets = 10,000
* Mean target speed = 0.7 m/s
* Std. dev. of target speed = 0.23 m/s
* = π/3 rad
* = 100 m
* # of sensors = 20, 50, 100
* Sensor Range: 10, 20, 30m
* Sensing Angle:
* # of Discretization’s: 6
* # of robots = 3
* Robot speed = 10 m/s
* = 1800s
* = 5400s

Figure 36: The number of targets found, by topologies of varying sensing models and sensing ranges, with a deployment start time of 1800s

Figure 37: The number of targets found, by topologies of varying sensing models and sensor numbers, with a deployment start time of 1800s

The topologies of each sensing model, for 50 sensors with a sensing range of 30m and a deployment start time of , are shown in Figure 38 below.

|  |  |
| --- | --- |
|  |  |
| a) | b) |
|  |  |
| c) | d) |

Figure 38: The topologies of the a) linear b) linear-directional c) directional and d) omni-directional networks

As expected, the best and worst performers were the omni-directional and linear networks respectively. The omni-directional model was able to detect more targets than the directional network in the best-case scenario. As this is not significantly higher than the other sensing models, the linear sensing model could also be used to plan the deployments of omni-directional sensors.

## Appendix K: Sensing Angle Study

The orientation of sensors in networks planned using linear sensors and directional sensors varies with increasing sensing angles. At sensing angles greater than , the sensors are oriented with the chord of the sector perpendicular to the LKP as a result of including a directional sensing model in the planning algorithm. This effect can be seen in the topologies presented in Figure 39. The parameters used to create these topologies are:

* # of Targets = 10,000
* Mean target speed = 0.7 m/s
* Std. dev. of target speed = 0.23 m/s
* = π/3 rad
* = 100 m
* # of sensors = 20
* Sensor Range: 100m
* Sensing Angle: , , ,
* # of Discretization’s: 6
* # of robots = 3
* Robot speed = 10 m/s
* = 1800 s
* = 5400 s

|  |  |
| --- | --- |
|  |  |
| a) | b) |
|  |  |
| c) | d) |
|  |  |
| e) | f) |
|  |  |
| g) | h) |

Figure 39: The topologies of the linear-directional and directional sensor networks with sensing angles of a-b) , c-d) , e-f) and g-h)

As mentioned above, it can be seen that at higher sensing angles, the orientations of the sensors start to vary between the linear-directional and directional sensors. This effect is most pronounced in Figure 39g and Figure 39h where the linear-directional topology has large gaps between sensors whereas the directional network more closely resembles a ring-of-fire type deployment. As a result, the sensor network in the latter case would intercept more targets and thus perform better as is shown in the analysis of Figure 18 in the simulations section of the paper.

## Appendix L: Graphs of Comparative Study

Additional results of the comparative study with a start and end time in the interval [are shown in Figure 40 and Figure 41 and for a start and end time in the interval ] in Figure 42 and Figure 43. The parameters used to create these results were:

* # of Targets = 10,000
* Mean target speed = 0.7 m/s
* Std. dev. of target speed = 0.23 m/s
* = π/3 rad
* = 100 m
* # of sensors = 20, 50, 100
* Sensor Range: 10, 20, 30m
* Sensing Angle:
* # of Discretization’s: 6
* # of robots = 3
* Robot speed = 10 m/s
* = 1800s, 7200 s
* = 5400s, 10800 s

Figure 40: The number of targets found by a set of 50 sensors with varying sensor ranges with a deployment start time of 1800s

Figure 41: The number of targets found by topologies of a varying number of sensors with a sensing range of 20m and a deployment start time of 1800s

Figure 42: The number of targets found by a set of 50 sensors with varying sensor ranges with a deployment start time of 7200s

Figure 43: The number of targets found by topologies of a varying number of sensors with a sensing range of 20m and a deployment start time of 7200s

# References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, “Wireless sensor networks: a survey,” *Comput. Netw.*, p. 30, 2002.

[2] M. Á. Serna, R. Casado, A. Bermúdez, N. Pereira, and S. Tennina, “Distributed Forest Fire Monitoring Using Wireless Sensor Networks,” *Int. J. Distrib. Sens. Netw.*, vol. 11, no. 10, p. 964564, Oct. 2015, doi: 10.1155/2015/964564.

[3] Y.-H. Xu, Q.-Y. Sun, and Y.-T. Xiao, “An Environmentally Aware Scheme of Wireless Sensor Networks for Forest Fire Monitoring and Detection,” *Future Internet*, vol. 10, no. 10, p. 102, Oct. 2018, doi: 10.3390/fi10100102.

[4] B. Kadri, B. Bouyeddou, and D. Moussaoui, “Early Fire Detection System Using Wireless Sensor Networks,” in *2018 International Conference on Applied Smart Systems (ICASS)*, Nov. 2018, pp. 1–4, doi: 10.1109/ICASS.2018.8651977.

[5] M. V. Ramesh, “Design, development, and deployment of a wireless sensor network for detection of landslides,” *Ad Hoc Netw.*, vol. 13, pp. 2–18, Feb. 2014, doi: 10.1016/j.adhoc.2012.09.002.

[6] X. Li, X. Zhang, and B. Guo, “Application of Collocated GPS and Seismic Sensors to Earthquake Monitoring and Early Warning,” *Sensors*, vol. 13, no. 11, pp. 14261–14276, Oct. 2013, doi: 10.3390/s131114261.

[7] Z. Sun, P. Wang, M. C. Vuran, M. A. Al-Rodhaan, A. M. Al-Dhelaan, and I. F. Akyildiz, “BorderSense: Border patrol through advanced wireless sensor networks,” *Ad Hoc Netw.*, vol. 9, no. 3, pp. 468–477, May 2011, doi: 10.1016/j.adhoc.2010.09.008.

[8] J. Cashbaugh and C. Kitts, “Optimizing Sensor Locations in a Multisensor Single-Object Tracking System,” *Int. J. Distrib. Sens. Netw.*, vol. 11, no. 7, p. 741491, Jul. 2015, doi: 10.1155/2015/741491.

[9] M. Akter, M. O. Rahman, M. N. Islam, M. M. Hassan, A. Alsanad, and A. K. Sangaiah, “Energy-Efficient Tracking and Localization of Objects in Wireless Sensor Networks,” *IEEE Access*, vol. 6, pp. 17165–17177, 2018, doi: 10.1109/ACCESS.2018.2809692.

[10] Y. Wen, R. Gao, and H. Zhao, “Energy Efficient Moving Target Tracking in Wireless Sensor Networks,” *Sensors*, vol. 16, no. 1, Jan. 2016, doi: 10.3390/s16010029.

[11] S. F. Ochoa and R. Santos, “Human-centric wireless sensor networks to improve information availability during urban search and rescue activities,” *Inf. Fusion*, vol. 22, pp. 71–84, Mar. 2015, doi: 10.1016/j.inffus.2013.05.009.

[12] M. Erdelj, M. Król, and E. Natalizio, “Wireless Sensor Networks and Multi-UAV systems for natural disaster management,” *Comput. Netw.*, vol. 124, pp. 72–86, Sep. 2017, doi: 10.1016/j.comnet.2017.05.021.

[13] J. Francisco, G.-H. Fernando, and L.-A. Carlos, “A Coordinated Wilderness Search and Rescue Technique Inspired by Bacterial Foraging Behavior,” in *2018 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, Dec. 2018, pp. 318–324, doi: 10.1109/ROBIO.2018.8665267.

[14] J. Chong Lee Shin, Z. Kashino, G. Nejat, and B. Benhabib, “A Sensor-Network-Supported Mobile-Agent-Search Strategy for Wilderness Rescue,” *Robotics*, vol. 8, no. 3, p. 61, Sep. 2019, doi: 10.3390/robotics8030061.

[15] M. Amac Guvensan and A. Gokhan Yavuz, “On coverage issues in directional sensor networks: A survey,” *Ad Hoc Netw.*, vol. 9, no. 7, pp. 1238–1255, Sep. 2011, doi: 10.1016/j.adhoc.2011.02.003.

[16] K. Plarre and P. R. Kumar, “Tracking objects with networked scattered directional sensors,” *EURASIP J. Adv. Signal Process.*, vol. 2008, p. 74:1–74:10, Jan. 2008, doi: 10.1155/2008/360912.

[17] M. A. Guvensan and A. G. Yavuz, “A New Coverage Improvement Algorithm Based on Motility Capability of Directional Sensor Nodes,” in *Ad-hoc, Mobile, and Wireless Networks*, 2011, pp. 206–219.

[18] C.-H. Wu and Y.-C. Chung, “A tiling-based approach for directional sensor network deployment,” in *2010 IEEE SENSORS*, Nov. 2010, pp. 1358–1363, doi: 10.1109/ICSENS.2010.5690541.

[19] C.-K. Liang, C.-H. Tsai, and M.-C. He, “On area coverage problems in directional sensor networks,” in *The International Conference on Information Networking 2011 (ICOIN2011)*, Jan. 2011, pp. 182–187, doi: 10.1109/ICOIN.2011.5723175.

[20] T.-W. Sung, Y.-T. Lu, F.-T. Lin, and C.-S. Yang, “Direction Control Using Delaunay Triangulation for Coverage Improvement in Directional Sensor Networks,” in *2015 Third International Conference on Robot, Vision and Signal Processing (RVSP)*, Nov. 2015, pp. 290–293, doi: 10.1109/RVSP.2015.76.

[21] T.-W. Sung and C.-S. Yang, “Distributed Voronoi-Based Self-Redeployment for Coverage Enhancement in a Mobile Directional Sensor Network,” *Int. J. Distrib. Sens. Netw.*, vol. 9, no. 11, p. 165498, Nov. 2013, doi: 10.1155/2013/165498.

[22] X. Han, X. Cao, E. L. Lloyd, and C.-C. Shen, “Deploying Directional Sensor Networks with Guaranteed Connectivity and Coverage,” in *2008 5th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, Jun. 2008, pp. 153–160, doi: 10.1109/SAHCN.2008.28.

[23] U.-R. Chen, B.-S. Chiou, J.-M. Chen, and W. Lin, “An Adjustable Target Coverage Method in Directional Sensor Networks,” in *2008 IEEE Asia-Pacific Services Computing Conference*, Dec. 2008, pp. 174–180, doi: 10.1109/APSCC.2008.37.

[24] V. Akbarzadeh, C. Gagne, M. Parizeau, M. Argany, and M. A. Mostafavi, “Probabilistic Sensing Model for Sensor Placement Optimization Based on Line-of-Sight Coverage,” *IEEE Trans. Instrum. Meas.*, vol. 62, no. 2, pp. 293–303, Feb. 2013, doi: 10.1109/TIM.2012.2214952.

[25] J. Wang, C. Niu, and R. Shen, “Priority-based target coverage in directional sensor networks using a genetic algorithm,” *Comput. Math. Appl.*, vol. 57, no. 11, pp. 1915–1922, Jun. 2009, doi: 10.1016/j.camwa.2008.10.019.

[26] P. Singh and S. Mini, “A Heuristic to Deploy Directional Sensor Nodes in Wireless Sensor Networks,” in *2016 8th International Conference on Computational Intelligence and Communication Networks (CICN)*, Dec. 2016, pp. 11–15, doi: 10.1109/CICN.2016.11.

[27] Y. E. Osais, M. St-Hilaire, and F. R. Yu, “Directional Sensor Placement with Optimal Sensing Range, Field of View and Orientation,” *Mob. Netw. Appl.*, vol. 15, no. 2, pp. 216–225, Apr. 2010, doi: 10.1007/s11036-009-0179-0.

[28] Z. Kashino, G. Nejat, and B. Benhabib, “A Hybrid Strategy for Target Search Using Static and Mobile Sensors,” *IEEE Trans. Cybern.*, pp. 1–13, 2019, doi: 10.1109/TCYB.2018.2875625.

[29] Z. Kashino, G. Nejat, and B. Benhabib, “A multi-robot sensor-delivery planning strategy for static-sensor networks,” in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2017, pp. 6640–6647, doi: 10.1109/IROS.2017.8206578.

[30] Y. Wang and C.-H. Wu, “Robot-Assisted Sensor Network Deployment and Data Collection,” in *2007 International Symposium on Computational Intelligence in Robotics and Automation*, Jun. 2007, pp. 467–472, doi: 10.1109/CIRA.2007.382919.

[31] L. C. Shiu, “The robot deployment scheme for wireless sensor networks in the concave region,” in *2009 International Conference on Networking, Sensing and Control*, Mar. 2009, pp. 581–586, doi: 10.1109/ICNSC.2009.4919342.

[32] G. Fletcher, X. Li, A. Nayak, and I. Stojmenovic, “Back-Tracking Based Sensor Deployment by a Robot Team,” in *2010 7th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, Jun. 2010, pp. 1–9, doi: 10.1109/SECON.2010.5508232.

[33] Y.-H. Wang, C.-H. Tsai, and Y.-H. Wu, “Robot-based deployment mechanism for wireless sensor networks in unknown region,” in *2013 International Joint Conference on Awareness Science and Technology Ubi-Media Computing (iCAST 2013 UMEDIA 2013)*, Nov. 2013, pp. 143–149, doi: 10.1109/ICAwST.2013.6765424.

[34] E. Mesa-Barrameda, N. Santoro, W. Shi, and N. Taleb, “Sensor deployment by a robot in an unknown orthogonal region: Achieving full coverage,” in *2014 20th IEEE International Conference on Parallel and Distributed Systems (ICPADS)*, Dec. 2014, pp. 951–960, doi: 10.1109/PADSW.2014.7097915.

[35] Rajesh M, A. George, and Sudarshan T.S.B., “Energy efficient deployment of Wireless Sensor Network by multiple mobile robots,” in *2015 International Conference on Computing and Network Communications (CoCoNet)*, Dec. 2015, pp. 72–78, doi: 10.1109/CoCoNet.2015.7411169.

[36] M. A. Batalin and G. S. Sukhatme, “Coverage, Exploration and Deployment by a Mobile Robot and Communication Network,” *Telecommun. Syst.*, vol. 26, no. 2, pp. 181–196, Jun. 2004, doi: 10.1023/B:TELS.0000029038.31947.d1.

[37] G. Tuna, V. C. Gungor, and K. Gulez, “An autonomous wireless sensor network deployment system using mobile robots for human existence detection in case of disasters,” *Ad Hoc Netw.*, vol. 13, pp. 54–68, Feb. 2014, doi: 10.1016/j.adhoc.2012.06.006.

[38] K. Woiceshyn, Z. Kashino, G. Nejat, and B. Benhabib, “Vehicle Routing for Resource Management in Time-Phased Deployment of Sensor Networks,” *IEEE Trans. Autom. Sci. Eng.*, vol. 16, no. 2, pp. 716–728, Apr. 2019, doi: 10.1109/TASE.2018.2857630.

[39] Z. Wang, X. Zhao, Z. Wang, and X. Qian, “Carrier-based sensor deployment by a mobile robot for wireless sensor networks,” in *2012 12th International Conference on Control Automation Robotics Vision (ICARCV)*, Dec. 2012, pp. 1663–1668, doi: 10.1109/ICARCV.2012.6485429.

[40] M. Varposhti, P. Saleh, S. Afzal, and M. Dehghan, “Distributed area coverage in mobile directional sensor networks,” in *2016 8th International Symposium on Telecommunications (IST)*, Sep. 2016, pp. 18–23, doi: 10.1109/IS℡.2016.7881773.

[41] H. Ma and Y. Liu, “On Coverage Problems of Directional Sensor Networks,” in *Mobile Ad-hoc and Sensor Networks*, Berlin, Heidelberg, 2005, pp. 721–731, doi: 10.1007/11599463\_70.

[42] A. A. Kumar and K. M. Sivalingam, “Target tracking in a WSN with directional sensors using electronic beam steering,” in *2012 Fourth International Conference on Communication Systems and Networks (COMSNETS 2012)*, Jan. 2012, pp. 1–10, doi: 10.1109/COMSNETS.2012.6151338.

[43] A. H. Mohajerzadeh, H. Jahedinia, Z. Izadi-Ghodousi, D. Abbasinezhad-Mood, and M. Salehi, “Efficient target tracking in directional sensor networks with selective target area’s coverage,” *Telecommun. Syst.*, vol. 68, no. 1, pp. 47–65, May 2018, doi: 10.1007/s11235-017-0373-5.

[44] A. Tripathi, H. P. Gupta, T. Dutta, D. Kumar, S. Jit, and K. K. Shukla, “A Target Tracking System Using Directional Nodes in Wireless Sensor Networks,” *IEEE Syst. J.*, vol. 13, no. 2, pp. 1618–1627, Jun. 2019, doi: 10.1109/JSYST.2018.2864684.

[45] Z. Wang, J. Liao, Q. Cao, H. Qi, and Z. Wang, “Barrier Coverage in Hybrid Directional Sensor Networks,” in *2013 IEEE 10th International Conference on Mobile Ad-Hoc and Sensor Systems*, Oct. 2013, pp. 222–230, doi: 10.1109/MASS.2013.12.

[46] L. Zhao, G. Bai, H. Shen, and Z. Tang, “Strong barrier coverage of directional sensor networks with mobile sensors,” *Int. J. Distrib. Sens. Netw.*, vol. 14, no. 2, p. 1550147718761582, Feb. 2018, doi: 10.1177/1550147718761582.

[47] T. Benahmed and K. Benahmed, “Optimal barrier coverage for critical area surveillance using wireless sensor networks,” *Int. J. Commun. Syst.*, vol. 32, no. 10, p. e3955, 2019, doi: 10.1002/dac.3955.

[48] X. Liu, B. Yang, and G. Chen, “Full-view barrier coverage in mobile camera sensor networks,” *Wirel. Netw.*, vol. 25, no. 8, pp. 4773–4784, Nov. 2019, doi: 10.1007/s11276-018-1764-6.

[49] Z. Kashino, J. Vilela, J. Y. Kim, G. Nejat, and B. Benhabib, “An adaptive static-sensor network deployment strategy for detecting mobile targets,” in *2016 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, Oct. 2016, pp. 1–8, doi: 10.1109/SSRR.2016.7784269.

[50] Z. Kashino, J. Y. Kim, G. Nejat, and B. Benhabib, “Spatiotemporal Adaptive Optimization of a Static-Sensor Network via a Non-Parametric Estimation of Target Location Likelihood,” *IEEE Sens. J.*, vol. 17, no. 5, pp. 1479–1492, Mar. 2017, doi: 10.1109/JSEN.2016.2638623.

[51] J. Vilela, Z. Kashino, R. Ly, G. Nejat, and B. Benhabib, “A Dynamic Approach to Sensor Network Deployment for Mobile-Target Detection in Unstructured, Expanding Search Areas,” *IEEE Sens. J.*, vol. 16, no. 11, pp. 4405–4417, Jun. 2016, doi: 10.1109/JSEN.2016.2537331.

[52] E. Yanmaz and H. Guclu, “Stationary and Mobile Target Detection Using Mobile Wireless Sensor Networks,” Apr. 2010, pp. 1–5, doi: 10.1109/INFCOMW.2010.5466620.

[53] Y. Zou and K. Chakrabarty, “Sensor deployment and target localization in distributed sensor networks,” *ACM Trans. Embed. Comput. Syst.*, vol. 3, no. 1, pp. 61–91, Feb. 2004, doi: 10.1145/972627.972631.

[54] R. J. Koester, *Lost Person Behavior: A Search and Rescue Guide on where to Look for Land, Air, and Water*. dbS Productions, 2008.

[55] H. R. Künsch, “Particle filters,” *Bernoulli*, vol. 19, no. 4, pp. 1391–1403, Sep. 2013, doi: 10.3150/12-BEJSP07.

[56] L. Lin and M. A. Goodrich, “A Bayesian approach to modeling lost person behaviors based on terrain features in Wilderness Search and Rescue,” *Comput. Math. Organ. Theory*, vol. 16, no. 3, pp. 300–323, Sep. 2010, doi: 10.1007/s10588-010-9066-2.

[57] R. Hooke and T. A. Jeeves, “`` Direct Search’’ Solution of Numerical and Statistical Problems,” *J. ACM*, vol. 8, no. 2, pp. 212–229, Apr. 1961, doi: 10.1145/321062.321069.

[58] G. Garai and B. B. Chaudhuri, “A distributed hierarchical genetic algorithm for efficient optimization and pattern matching,” *Pattern Recognit.*, vol. 40, no. 1, pp. 212–228, Jan. 2007, doi: 10.1016/j.patcog.2006.04.023.

[59] J. C. Lagarias, J. A. Reeds, M. H. Wright, and P. E. Wright, “Convergence Properties of the Nelder-Mead Simplex Method in Low Dimensions,” *SIAM J. Optim.*, vol. 9, pp. 112–147, 1998.

[60] E. S. Biagioni and G. Sasaki, “Wireless sensor placement for reliable and efficient data collection,” in *36th Annual Hawaii International Conference on System Sciences, 2003. Proceedings of the*, Jan. 2003, p. 10 pp.-, doi: 10.1109/HICSS.2003.1174290.

[61] Y. Zou and Krishnendu Chakrabarty, “Sensor deployment and target localization based on virtual forces,” in *IEEE INFOCOM 2003. Twenty-second Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE Cat. No.03CH37428)*, Mar. 2003, vol. 2, pp. 1293–1303 vol.2, doi: 10.1109/INFCOM.2003.1208965.

[62] M. Rout and R. Roy, “Dynamic deployment of randomly deployed mobile sensor nodes in the presence of obstacles,” *Ad Hoc Netw.*, vol. 46, pp. 12–22, 2016, doi: 10.1016/j.adhoc.2016.03.004.

[63] S. P. Han, “A globally convergent method for nonlinear programming,” *J. Optim. Theory Appl.*, vol. 22, no. 3, pp. 297–309, Jul. 1977, doi: 10.1007/BF00932858.

[64] G. Tan, S. A. Jarvis, and A.-M. Kermarrec, “Connectivity-Guaranteed and Obstacle-Adaptive Deployment Schemes for Mobile Sensor Networks,” in *2008 The 28th International Conference on Distributed Computing Systems*, Jun. 2008, pp. 429–437, doi: 10.1109/ICDCS.2008.49.

[65] Z. Yun, X. Bai, D. Xuan, T. H. Lai, and W. Jia, “Optimal Deployment Patterns for Full Coverage and $k$-Connectivity $(k \leq 6)$ Wireless Sensor Networks,” *IEEEACM Trans. Netw.*, vol. 18, no. 3, pp. 934–947, Jun. 2010, doi: 10.1109/TNET.2010.2040191.

[66] H. P. Gupta, P. K. Tyagi, and M. P. Singh, “Regular Node Deployment for $k$ -Coverage in $m$ -Connected Wireless Networks,” *IEEE Sens. J.*, vol. 15, no. 12, pp. 7126–7134, Dec. 2015, doi: 10.1109/JSEN.2015.2471837.

[67] K. Sakai, M.-T. Sun, W.-S. Ku, T. H. Lai, and A. V. Vasilakos, “A Framework for the Optimal $k$ -Coverage Deployment Patterns of Wireless Sensors,” *IEEE Sens. J.*, vol. 15, no. 12, pp. 7273–7283, Dec. 2015, doi: 10.1109/JSEN.2015.2474711.

[68] E. D. Jou, “Determine Whether Two Line Segments Intersect,” in *Modeling in Computer Graphics: Proceedings of the IFIP WG 5.10 Working Conference Tokyo, Japan, April 8–12, 1991*, T. L. Kunii, Ed. Tokyo: Springer Japan, 1991, pp. 265–274.