Research Article

Traffic monitoring using an adaptive sensor power scheduling algorithm

Richard Tatum[1] · Matthew Bays[1] · John Hyland[1] · Benjamin Hartman[1]

Received: 3 June 2019 / Accepted: 12 October 2019 / Published online: 5 November 2019

Abstract

Using sensors to monitor surface or subsurface traffic requires sensor placement, detection of traffic changes, and sensor power scheduling for improved efficiency. Of these capabilities, sensor power scheduling is one of the most important as the appropriate sensors must be selected for activation to respond to changes in the traffic. We present an adaptive power scheduling algorithm that uses the homogeneous equilibrium of a potential-field-based dynamical system to determine which sensors should be active. Our algorithm assumes a nearest neighbor topology, which makes additional assumptions about the placement of sensors. We formalize these conditions and construct a sensor placement algorithm to support our scheduling algorithm. To demonstrate the efficacy of our scheduling approach, we provide two distinctive traffic detection algorithms that we combine with our placement and scheduling algorithm to test via simulation. We provide the simulation results that show in both cases, the adaptive scheduling algorithm behaves efficiently as compared to an area coverage approach , as well as an all-active path coverage approach.

Keywords Autonomous power scheduling · Traffic monitor · Nearest neighbor topology · Mixed-integer linear programming (MILP) · Potential field

1 Introduction

The idea of using statically placed sensors to monitor geographical regions is well understood and has been applied to monitoring marine species [2], underwater vehicles [1], or more specifically, port security [16, 18]. Three facets to the problem of traffic monitoring are where to place the sensors, how to detect traffic distribution changes, and how to schedule sensors in response to traffic distribution changes. With respect to sensor placement, this problem is either solved as an area coverage problem [6, 7, 11, 19] or path coverage problem [14, 20]. As for the remaining facets, McIntyre and Hintz use information gain to determine sensor schedules for searching for unknown entities [17]. Others have considered the trade-off between accuracy and power with regard to scheduling sensors for tracking individual entities [12, 23]. Our scheduling approach, which we discuss in the next paragraph, considers detections from multiple entities.

Our main contribution to the traffic monitoring problem is to focus on the scheduling facet of traffic monitoring. Simply stated, when a traffic distribution change is detected, a sensor scheduling algorithm must determine which sensors should be active and which should be inactive. Our approach, which uses statically placed sensors, borrows ideas from robotic path planning for mobile sensors [3, 15]. For mobile path planning, the velocity of the vehicle is proportional to the gradient of some potential function. Analogously, our approach assumes that the velocity of activating sensors is proportional to the gradient of an attractive potential field. Solving the resulting ordinary differential equation (ODE) for mobile path

✉ Richard Tatum, richard.d.tatum@navy.mil | [1]Naval Surface Warfare Center, Panama City Division, 110 Vernon Ave, Panama City, FL 32407, USA.

SN Applied Sciences (2019) 1:1552 | https://doi.org/10.1007/s42452-019-1494-0

planning yields a path, while the solution of our ODE yields the indices of the sensors to be active. Thus, our approach combines elements of static and mobile sensors to create a new, unique sensor scheduling method.

However, if sensors are not present where the traffic has shifted, then no scheduling of sensors can satisfactorily monitor the traffic. Thus, sensor placement clearly impacts the ability of any sensor schedule algorithm. In this paper, we more formally state placement assumptions and use them to construct a sensor placement algorithm to support our adaptive scheduling algorithm. In particular, our sensor placement algorithm uses a nearest neighbor topology, which is a modification of [20]; our placement approach considers the placement of only sensors.

Sensor scheduling begins with detections of changes to traffic distributions. Without such indications, the sensor scheduling algorithm cannot proceed. Therefore, we provide two unique traffic change detection algorithms that we now describe.

Our first traffic change detection approach relies on the following assumption: If traffic continuously shifts away from active sensors, then the total number of detections, which are the number of detections made by all active sensors, should decrease. To identify this event, we first use cubic b-splines as in [21] to smooth the noisy data of total detections. Using a uniform knot sequence, which guarantees C^2 smoothness of the b-spline everywhere over its domain [9], we are able to compute all of the inflection points and choose the inflection point with the highest number of detections. If the number of detections drops below the number of detections associated with the inflection point, we label that event as a traffic change event.

Our second detection approach identifies a traffic change event as one in which the Hellinger distance between the previous and current detection distributions meets a specified threshold. The idea of using the Hellinger distance to compute the distance between distributions is an accepted technique as found in [13], in which the authors apply the Hellinger distance to distinguish distributions for classification problems.

We combine our adaptive scheduling algorithm with the sensor placement approach and two traffic detection algorithms to form two completely distinctive traffic monitoring solutions. Figure 1 shows the conceptual behavior of these two traffic monitoring solutions. We compare these two methods with an area coverage approach found in [11] and an all-active path coverage approach found in [20] using a Monte Carlo simulation. In particular, we compare the probability of detections and the efficiency detections of all four monitoring solutions.

Our paper is organized as follows. Section 2 describes the symbols that we use in this paper. Section 3 describes assumptions of our approach, while Sect. 4 describes the adaptive scheduling algorithm itself. In Sect. 5, we use the assumptions of 3 to construct a sensor placement algorithm. We describe two traffic detection algorithms in 6. In Sect. 7, we include the simulation results and conclude with a summary in Sect. 8.

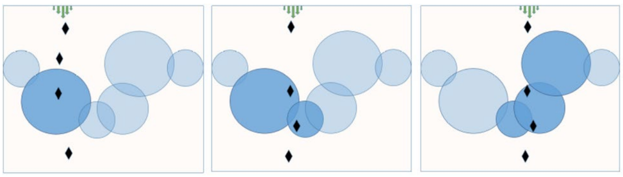

Fig. 1 Ships (black diamonds) are generated from the traffic distribution (green arrows). The ships move from top to bottom, passing over sensors, which are either active (transparent blue circles) or not active (opaque blue circles)

2 Nomenclature

B	Region in $\mathbb{R}^3$ to be monitored
$\mathscr{E}$	Entry boundary of traffic into the B
$\mathscr{X}$	Exit boundary of traffic from the B
$\mathscr{L}$	Left boundary of the B
$\mathscr{R}$	Right boundary of the B
$\boldsymbol{\tau}$	Element of B
$\boldsymbol{p_i}$	Position of i-th sensor
E_{env}	m–vector representing the environment
$\mathbf{e}$	Environmental vector
ρ_i	i-th sensor detection function
R_i	Maximum sensing range of the i-th sensor
$\mathscr{S}_i$	Sensing volume of the i-th sensor
M	Number of sensors
$\mathcal{A}$	Arrival traffic distribution
$T(\boldsymbol{\tau})$	Traffic trajectory of a single entrant into B
s_ξ	Sensor with largest number of detections
$s_{\xi+w}$	Rightmost active sensor
$x_{\xi+w}$	Index of rightmost active sensor
$s_{\xi-w}$	Leftmost active sensor
$x_{\xi-w}$	Index of leftmost active sensor
U	Arbitrary subset of $\mathbb{R}^n$
f	Potential field
$C^2(U)$	Space of twice continuously differentiable functions over U
g	Function belonging to $C^2(U)$
$\boldsymbol{x}^*$	Desired goal of potential field
κ	Potential field proportionality constant
$G(\boldsymbol{x})$	Jacobian of g
d_i	Number of detections for a sensor s_i
ν_i	Number of iterations with no detections for sensor s_i in active mode

v	Maximum of v_i
ζ_i	Number of iterations with no detections for a sensor s_i in standby mode
ζ	Maximum of ζ_i
d	Depth of traffic
$\mathscr{P}$	Plane of traffic at depth d
$\mathscr{S}'_i$	Intersection of $\mathscr{S}_i$ and $\mathscr{P}$.
β	Intersection of $\mathscr{E}$ and $\mathscr{P}$
$\mathcal{P}_\beta$	Orthogonal projector of planar sensing regions onto β
ψ_i	Projection of $\mathscr{S}'$ onto β
I_k	Domain of ψ_k
$\lvert I_k \rvert$	Length of interval I_k
α_k	Binary variable that indicates if elements of I_k belong to composite interval
$I_{\alpha_1,\ldots,\alpha_M}$	Composite interval
L_j	Set containing intervals formed from the overlap of j intervals
F	Objective function that estimates the probability of detection of all placed sensors
a	Constant for uniform traffic distribution
F_j	F evaluated over L_j
T	Triangular mesh representing environmental effects on performance
T_{jk}	Triangle belonging to T
N_T	Number of triangles in T
(x_i, y_i)	Planar position of sensor s_i
b_{ijk}	Binary variable that indicates if (x_i, y_i) belongs to T_{jk}
(x_{ijk}, y_{ijk})	Contribution from triangle T_{jk} to (x_i, y_i)
X_{jk}^{min}	Minimum value of all x_{ijk}
X_{jk}^{max}	Maximum value of all x_{ijk}
Y_{jk}^{min}	Minimum value of all y_{ijk}
Y_{jk}^{max}	Maximum value of all y_{ijk}
α_{jk}	x-Coefficient describing the hypotenuse of T_{jk}
β_{jk}	y-Coefficient describing the hypotenuse of T_{jk}
γ_{jk}	Constant describing the hypotenuse of T_{jk}
A_{ijk}	Constant describing sensor performance
B_{ijk}	x-Coefficient describing sensor performance
C_{ijk}	y-Coefficient describing sensor performance
$d_1()$	L_1-metric
R	Sensor radius
M_{max}	Maximum difference of all $y \in B$
$\delta^+_{i,i+1}$	$y_{i+1} - y_i$, if $y_{i+1} \geq y_i$
$\delta^-_{i,i+1}$	$y_i - y_{i+1}$, if $y_{i+1} < y_i$
$g_j(x_i, y_i)$	General linear constraint function
N_d	Sequence of detections from discrete time events d_1 to d_N
$\mathcal{D}$	Smooth approximation of N_d
S	Set of inflection points of $\mathcal{D}$
p^*	Time event for which the number of detections is maximal
h	Distribution of detections over all sensors
$\delta^{(k)}$	Set of new detections over all sensors at k−th time step
$h^{(k)}$	Aggregated distribution of detections
$d^2(h, h^{(k)})$	Hellinger distance
α	Order of Hellinger distance
$U[x, x+2]$	Uniform random distribution defined over $[x, x+2]$
υ	Homotopy parameter that varies influence between a static and dynamic uniform random distribution
e	Detection efficiency
N_a	Total amount of active time
N_t	Total number of ships
N_a	Total number of active time units

3 Assumptions

The overall goal of our adaptive scheduling algorithm is to use statically placed sensors in some volume $B \subset \mathbb{R}^3$ to adjust the sensor modes to monitor the movement of the traffic distribution. We now describe our sensor, field, and traffic distribution assumptions.

3.1 Sensors and field

We identify the sensor modes as *active, standby,* and *off.* Our algorithm explicitly decides which sensors should be placed in the active state, thus allowing a sensor to make detections. Once active, the sensor remains active as long as it receives detections within v time units or is explicitly selected again by the scheduling algorithm. However, if no detections are received within v time units, the sensor shifts to the standby state. Once in the standby state, if the sensor receives detections, it can again be placed in the active state. If no detections are received for ζ time units, the sensor is placed in the off state, in which it cannot receive any detections. However, once in the off state, a sensor cannot be placed in the active state again without being explicitly chosen by the scheduling algorithm.

For $B = [0, d_x] \times [0, d_y] \times [0, d_z]$, we are assuming that traffic enters a single face of B denoted as $\mathscr{E}$ and exits a different face denoted as $\mathscr{X}$. Let us assume for $\boldsymbol{\tau} = (\tau_x, \tau_y, \tau_z)$, the entrance face is given as $\mathscr{E} = \{\boldsymbol{\tau} \in B : \tau_y = d_y\}$ and the exit face is given as $\mathscr{X} = \{\boldsymbol{\tau} \in B : \tau_y = 0\}$. Furthermore, we label the left face as $\mathscr{L} = \{\boldsymbol{\tau} \in B : \tau_x = 0\}$ and the right face as $\mathscr{R} = \{\boldsymbol{\tau} \in B : \tau_x = d_x\}$.

Using the field B, we define our sensors in the following way

Definition 1 (Sensor) For sensor position

$$\boldsymbol{p_i} = (x_i, y_i, d_i) \in B, \tag{1}$$

environmental vector $\boldsymbol{e} \in E_{env} \subset R^m$, and $\boldsymbol{\tau} \in B$, the probability of detection at $\boldsymbol{\tau}$ is the spherically symmetric $\rho_i(\boldsymbol{p_i}, \mathbf{e}, \boldsymbol{\tau})$ where $\rho_i : B \times E_{env} \times B \rightarrow (0, 1]$. The i−th sensing volume is defined as

$$\mathscr{S}_i = \left\{\boldsymbol{\tau} \in B : \rho_i(\boldsymbol{p_i}, \boldsymbol{e}, \boldsymbol{\tau}) > 0, \left\|\boldsymbol{p_i} - \boldsymbol{\tau}\right\|_2 \leq R_i\right\}, \tag{2}$$

where radius R_i is the maximum range of the i−th sensor. We drop the explicit mention of $\boldsymbol{p_i}$ and $\boldsymbol{e}$ for ease of exposition and refer to $\rho_i(\boldsymbol{\tau})$.

3.2 Traffic

The arrival distribution is defined as $\mathcal{A}(\boldsymbol{\tau}) : \mathscr{E} \rightarrow [0, 1]$. We define a traffic trajectory as a set of points $T(\boldsymbol{\tau}) \subset B$ with initial point $\boldsymbol{\tau_0} \in \mathscr{E}$ that intersects $\mathscr{X}$, but does not intersect $\mathscr{L}$ and $\mathscr{R}$. For this type of traffic scenario, we require

$$T(\boldsymbol{\tau}) \cap \mathscr{S}_j \neq \emptyset \text{ for all } T(\boldsymbol{\tau}) \text{ with } \boldsymbol{\tau_0} \in \mathscr{E}. \tag{3}$$

In other words, we want to avoid spatial gaps in coverage of the paths that traverse B, which implies sensors must be placed so that every trajectory has a nonzero probability of intersecting at least one sensor region in the field.

The set of configurations that satisfy (3) would include both area coverage and path coverage algorithms. We consider path coverage algorithms, and in particular, we focus on path coverage as provided by the nearest neighbor topology as it seems to provide adequate coverage of the traffic, but with a smaller number of sensors. We confirm this idea with our simulation results that we present later.

The configuration conditions are given as

$$\mathscr{S}_i \cap \mathscr{S}_{i+1} \neq \emptyset \text{ for } 1 \leq i \leq M, \tag{4}$$

$$\mathscr{S}_i \cap \mathscr{S}_j = \emptyset \text{ for } j \neq i-1, i, i+1, \tag{5}$$

$$\mathscr{S}_1 \cap \mathscr{L} \neq \emptyset, \tag{6}$$

$$\mathscr{S}_M \cap \mathscr{R} \neq \emptyset. \tag{7}$$

Primarily, (4) and (5) establish a nearest neighbor type of topology. Inequalities (6) and (7) are boundary conditions that ensure complete sensor coverage near $\mathscr{L}$ and $\mathscr{R}$. Since we consider a nearest neighbor topological configuration that satisfies (3), decreases in traffic through active sensor regions implies that the traffic has shifted to one of the inactive sensors. Note that with the nearest neighbor topology, if a traffic distribution change event occurs, our scheduling algorithm needs only to consider two possible search directions.

We assume that the sensors will be able to communicate while not necessarily powered on with regard to sensing. Note that the nearest neighbor sensor topology also has the benefit of supporting energy-efficient communication peer-to-peer queries [10].

4 Adaptive power scheduling algorithm

Now we describe our scheduling algorithm. Our algorithm begins with all sensors set to their active state. We then identify s_ξ, which is the sensor with the largest number of detections and ensure that $s_{\xi-w}, \cdots s_\xi, \ldots s_{\xi+w}$, are active and $x_{\xi-w}, \ldots x_\xi, \ldots x_{\xi+w}$, are their corresponding sensor index values. As the traffic changes, we expect the sensor s_ξ to also change. To determine which sensors should be selected for active state, we can solve the following

$$\dot{x} = \nabla f(x) \tag{8}$$

over the space of sensor indices for some attractive potential function f, which has been used for robotic path planning [3, 15]. However, rather than path planning for mobile sensors, we use the solutions for this ODE to determine which static sensors should scheduled as active.

Although Eq. (8) is a single ODE, we consider higher dimensional potential functions of the form

$$f(\mathbf{x}) = g(\kappa\|\mathbf{x} - \mathbf{x}^*\|_2^2),$$

with $\kappa > 0$ and $\mathbf{x}^*$ is the desired goal, and show the choice of the rest point, namely, $\mathbf{x}^*$, is the asymptotic or long term solution for $\mathbf{x_0} \in \mathbb{R}^n$.

Proposition 1 *For $\mathbf{x}^* \in U \subset \mathbb{R}^n$, suppose that for $g \in C^2(U)$, $\frac{\partial g}{\partial x_i}$ is a strictly decreasing function for all $1 \leq i \leq n$. For $\dot{\mathbf{x}} = \nabla g(\kappa\|\mathbf{x} - \mathbf{x}^*\|_2^2)$ the rest point $\mathbf{x}^*$ is asymptotically stable.*

Proof A simple computation shows

$$\dot{\mathbf{x}} = 2\kappa G(\mathbf{x})(\mathbf{x} - \mathbf{x}^*)$$

where

$$G(\mathbf{x}) = \begin{bmatrix} \frac{\partial g(\kappa\|\mathbf{x}-\mathbf{x}^*\|_2^2)}{\partial x_1} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \frac{\partial g(\kappa\|\mathbf{x}-\mathbf{x}^*\|_2^2)}{\partial x_n} \end{bmatrix}.$$

For $\mathbf{x_0} \in U$, solutions exist and are unique as the right hand side is continuously differentiable and consequently, Lipschitz continuous on the neighborhood U. By the Cauchy-Lipschitz existence and uniqueness theorem, the system will have unique solutions with initial conditions $\mathbf{x_0} \in U$. The linearization of the right hand side yields

$$\dot{\mathbf{x}} = 2\kappa G(\mathbf{x}^*)\mathbf{x}, \tag{9}$$

which has all negative eigenvalues, since $\frac{\partial g}{\partial x_i} < 0$ for all $1 \leq i \leq n$. By the spectral stability theorem [8], we see that $\mathbf{x}^*$ is asymptotically stable. □

Proposition (1) shows that under a small number of assumptions, the model governed by Eq. (8) will produce solutions that in the limit, will converge to $x^* = x_{max}$ in a sufficiently small neighborhood, where x_{max} is the index corresponding to the sensor with the most detections. Therefore, when a traffic change is detected, we can assign $x_\xi = x_{max}$, as x_{max} is the asymptotic solution for Eq. (8) when f is of the form $g(\kappa\|x - x_{max}\|_2^2)$. Algorithm (1) uses the asymptotically stable rest point for scheduling and shows the major steps of the scheduling algorithm.

Algorithm 1 Sensor Power Schedule Algorithm

```
Require: Initially set $s_i$ mode to active for all $1 \leq i \leq M$

if A change in the $\mathcal{A}$ is detected then
    $x_{max} = \max_{x_i} d_i$, where $d_i$ is the number of detections of sensor $s_i$
    $x_\xi \leftarrow x_{max}$

    if $i \in [x_\xi - w, x_\xi + w]$ then
        Set $s_i$ mode to active
    end if

end if

if $s_i$ is in active mode then

    if $d_i = 0$ then
        $\nu_i \leftarrow \nu_i + 1$

        if $\nu_i = \nu$ then
            Set $s_i$ mode to standby
            $\nu_i = 0$
        end if

    else
        $\nu_i = 0$
    end if

    if $s_i$ is in standby mode then

        if $d_i = 0$ then
            $\zeta_i \leftarrow \zeta_i + 1$

            if $\zeta_i = \zeta$ then
                Set $s_i$ mode to off
                $\zeta_i = 0$
            end if

        else
            Set $s_i$ mode to active
            $\zeta_i = 0$
        end if

    end if

end if

if $s_i$ is in off mode for all $1 \leq i \leq M$ then
    Set $s_i$ to active for all $1 \leq i \leq M$.
    $w \leftarrow w + 2$
end if
```

As earlier stated, we now describe a placement algorithm that adheres to the conditions described by (4)–(3). This algorithm is later used to demonstrate the utility of our adaptive scheduling algorithm.

5 Sensor placement

The purpose of this section is to demonstrate how to create a placement algorithm that adheres to the constraints of our adaptive algorithm. Thus, we use the coverage Condition (3) to create an objective function that we constrain using Conditions (4)–(7) to determine the placement of sensors. We consider approaches for both homogeneous and heterogeneous environments.

5.1 Objective function

In particular, the path coverage Condition (3) is satisfied for the case in which traffic moves through B at some depth d on a fixed plane $\mathscr{P}$. Observe that while the depth of the traffic is fixed, the distance from bottomed sensors to the depth is not constant, as found in [20, 22]. In fact, a fixed depth gives a design variable that can be adjusted for various scenarios, such as monitoring surface traffic, $d = 0$ or subsurface traffic, $d < 0$. Thus, we consider this type of scenario and define two dimensional sensor regions as

$$\mathscr{S}'_i = \mathscr{S}_i \cap \mathscr{P}, \tag{10}$$

centered at (x_i, y_i, d). We define the entrance boundary as $\beta = \mathscr{E} \cap \mathscr{P}$ and define the arrival distribution $\mathcal{A}(x) : \beta \rightarrow [0, 1]$.

Assuming that traffic moves in an orthogonal direction relative to β, for fixed $x \in \beta$ and depth d, we use the orthogonal projector $\mathcal{P}_\beta$ to compute

$$\psi_i(x) \triangleq \mathcal{P}_\beta(\mathscr{S}'_i) = \max_{\boldsymbol{\tau}=(x,y,d)\in\mathscr{S}'_i} \rho_i(\boldsymbol{\tau}), \tag{11}$$

which represents the maximum probability of detection at $x \in \beta$. Assuming no overlap of M sensors, we compute the maximum likelihood of detecting arriving traffic as

$$F = \sum_{i=1}^{M} \int_{I_i} \mathcal{A}(x)\psi_i(x)dx. \tag{12}$$

Since we are unlikely to know the exact distribution of $\mathcal{A}$ or all a priori changes, we will assume that $\mathcal{A}$ is a uniform distribution and write

$$F = a \sum_{i=1}^{M} \int_{I_i} \psi_i(x)dx. \tag{13}$$

Effectively, all positions are equally likely as the minimum variance solution, which will prevent the placement algorithm from ignoring coverage gaps. Note that this formulation of F assumes that there is no overlap of the sensors. To account for the overlap, we identify those intervals that are formed by overlapping intervals as $I_{\alpha_1,\cdots,\alpha_M}$ where

$$\alpha_k = \begin{cases} 1 \text{ if } x \in I_k \\ 0 \text{ otherwise} \end{cases} \tag{14}$$

and write the set containing intervals formed from the intersection of j intervals as

$$L_j = \{I_{\alpha_1,\ldots,\alpha_M} : \sum_{i=1}^{M} \alpha_i = j\}. \tag{15}$$

Using L_j we partition our objective function as

$$F = a \sum_j F_j, \tag{16}$$

where

$$F_j = \sum_{I \in L_j} \int_I 1 - \prod_{i=1}^{M} (1 - \psi_i(x))^{\alpha_i} dx, \tag{17}$$

We now consider the objective function in both homogeneous and heterogeneous environments for the placement of sensors.

5.2 Homogeneous environment

In a homogeneous environment, $\psi_i(x)$ remains constant with respect to traffic at a particular depth. Furthermore, the length of the interval I_i is given as $|I_i| = l$ for all ψ_i. Consequently, all of the ψ_i are simple linear translations of each other. The next proposition shows that if we only place two sensors, and $|I_1| + |I_2| \leq |\beta|$, then the two sensors must be placed such that $I_1 \cap I_2 = \emptyset$ to achieve an optimal placement.

Proposition 2 *For an area $B = [0, d_x] \times [0, d_y] \times [0, d_z] \subset \mathbb{R}^3$ and homogeneous environment E, suppose that for two sensors (as defined in Definition (1) with corresponding intervals I_i and I_j,*

$$|I_i| + |I_j| \leq |\beta|.$$

The objective function F as defined in Eq. (17) is maximized if and only if $\text{int}(I_i) \cap \text{int}(I_j) = \emptyset$, *where int represents the interior of the interval.*

Proof Consider two sensors, s_i and s_j and a uniform environment E. We can write F as

$$F = a\left(\int_{I_i} \psi_i(x)dx + \int_{I_j} \psi_j(x)dx + \int_{I_i \cap I_j} \psi'(x)dx\right).$$

Comparing the computation of F for the cases of interval overlap and no interval overlap reveals the F will only attain its maximum value if and only if there is no overlap. □

This proposition guarantees optimal placement as long as the sensors do not overlap. Satisfaction of the nearest neighbor topology Conditions (4)–(7) imply necessary optimality if the sensors overlap only the boundaries. In other words, no complex planning technique is required for placement for this case. However, for the situation in which Conditions (4)–(7) can only be satisfied with overlapping sensors, or the environment is heterogeneous, we resort to more tractable computational methods.

5.3 Heterogeneous environment

We consider the case in which sensor performance varies non-trivially as a function of a heterogeneous environment, as in Fig. 2. This is the type of environment found in [14], where the sensor performance varies as function of the environment with an assumed constant sensor radius R. We now describe how to represent the sensor performance for such environments, as well as how to implement the configuration conditions given by (4)–(7).

We capture the effects of the environment on the sensor by using a triangular mesh and define our sensor locations using such a mesh. Let (x_i, y_i) denote the positions of the i–th sensor and for each triangle $T_{jk} \in T$, where T is the triangular mesh with $(N_T + 1)^2$ grid points and $0 \leq j, k \leq N_T$. We represent the three sides of each triangle with the following constraints $\forall 1 \leq i \leq M, 0 \leq j, k \leq N_T$,

$$X^{min}_{jk} b_{ijk} \leq x_{ijk} \leq X^{max}_{jk} b_{ijk}, \tag{18}$$

$$Y^{min}_{jk} b_{ijk} \leq y_{ijk} \leq Y^{max}_{jk} b_{ijk}, \tag{19}$$

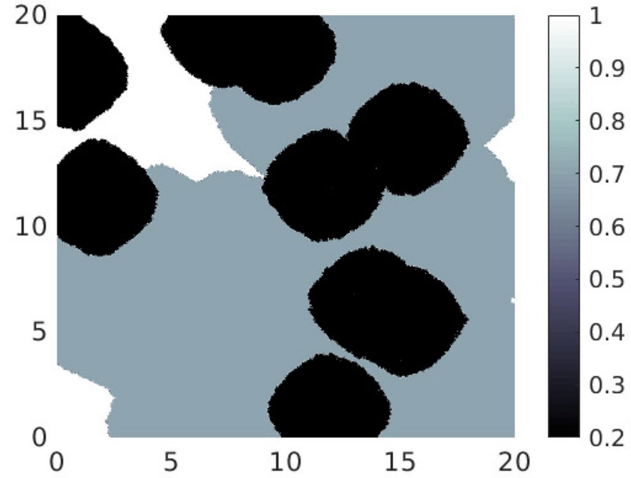

Fig. 2 Sensor performance map. The regions represent the probability of detection as a function of the environment

$$\alpha_{jk} x_{ijk} + \beta_{jk} y_{ijk} \{\leq, \geq\} \gamma_{jk}, \tag{20}$$

where Constraint (18) gives the bounds for x_{ijk}, Constraint (19) gives the bounds for y_{ijk}, Constraint (20) is the hypotenuse constraint for the each triangle T_{jk}, and $\alpha_{jk}, \beta_{jk}, \gamma_{jk}$ are the coefficients of the hypotenuse of T_{jk}. The variables (x_{ijk}, y_{ijk}) represent potential locations of the i–th sensor in T_{jk}. To ensure that each sensor has a unique position, we impose the additional constraints

$$x_i = \sum_{j,k} x_{ijk}, \tag{21}$$

$$y_i = \sum_{j,k} y_{ijk}, \tag{22}$$

$$\sum_{j,k} b_{ijk} = 1, \tag{23}$$

where Constraints (21) and (22) sum over all of the x_{ijk} and y_{ijk}, respectively, and we are guaranteed that

$(x_{ijk}, y_{ijk}) \in T_{ijk}$

by Constraint (23), since b_{ijk} is a binary variable that equals 1 when the point is within the triangle T_{jk} and 0 otherwise. Furthermore, summing all of the x_{ijk} and y_{ijk} yields the position of the i–th sensor (x_i, y_i).

We represent the sensor performance $p_i(x_i, y_i)$ over the triangular mesh T as linear approximations given by the following constraints

$$\begin{aligned} &\forall 1 \leq i \leq M, 0 \leq j, k \leq N_T, \\ &p_{ijk}(x_i, y_i) = A_{ijk} b_{ijk} - B_{ijk} x_i - C_{ijk} y_i \end{aligned} \tag{24}$$

$$p_i(x_i, y_i) = \sum_{jk} p_{ijk}(x_i, y_i), \tag{25}$$

where A_{ijk}, B_{ijk}, and C_{ijk}, are the coefficients that describe the sensor performance function $p_i(x_i, y_i)$ over $T_{ijk} \in T$.

As previously discussed, sensors are placed to minimize the risk of missed detections. Thus, the nearest neighbor and boundary conditions defined by (4)–(7) are converted to integer constraints as $\forall 1 \leq i \leq M - 1$,

$$x_i \leq x_{i+1}, \tag{26}$$

$$d_1(p_i, p_{i+1}) = x_{i+1} - x_i + |y_{i+1} - y_i|, \tag{27}$$

$$d_1(p_i, p_{i+1}) = 2R, \tag{28}$$

$$|y_{i+1} - y_i| = \delta^+_{i,i+1} - \delta^-_{i,i+1}, \tag{29}$$

$$0 \leq \delta^+_{i,i+1} \leq b_i M_{max}, \tag{30}$$

$$0 \leq \delta^-_{i,i+1} \leq (1 - b_i) M_{max}, \tag{31}$$

$$x_1 - R = 0 \tag{32}$$

$$x_M + R = |\beta|, \tag{33}$$

where $p_i = (x_i, y_i)$, and $M_{max} = \max_{y,y' \in B}(y - y')$. Constraint (26) imposes a consecutive ordering on the sensors. Constraint (27) defines the distance between consecutive sensors and (28) ensures that the sensors are placed so as to reduce overlap between them while maintaining complete coverage of incoming traffic. Constraint (29) defines the absolute value of the difference between the consecutive y values. However, to support this representation, we need Constraints (30) and (31) so as to define the positive and negative values of differences in consecutive y values, respectively. The indicator variable b_i is 1 if the difference is positive, and 0 otherwise. Constraint (32) ensures that leftmost edge of the first sensor begins at the left side of β, while Constraint (33) ensures that the rightmost edge of the last sensor coincides with the right side of β. Note that all of these constraints implement the desired configuration conditions specified in (4)–(7).

We modify the objective function described in 17 to reduce the computational complexity that will be problematic for this MILP problem. For our objective function, we only need support from L_0 and L_1, meaning that our objective function has the form of $F = F_0 + F_1$. However, we approximate F with only the first term, F_0, and get

$$F \approx 2Ra \sum_i p_i(x_i, y_i). \tag{34}$$

As such, we want to find the smallest M so as to

$$\begin{aligned} &\underset{(x,y) \in B}{\text{maximize}} && \sum_{i=1}^{M} p_i(x_i, y_i) \\ &\text{subject to} && g_j(x_i, y_i)\{=, \leq\} b_j, j = 1 \ldots m, \end{aligned} \tag{35}$$

where $g_j(x_i, y_i)$ are linear constraint functions defined in this section. To find the smallest M, we iterate from the smallest number of allowable sensors, and solve each problem until we find the set of sensors that maximizes the objective function while satisfying all of the constraints. Observe while the objective function seems relatively simple, we have $MN_T + M - 1$ binary variables for M sensors and N_T triangles in the mesh. As such, we employed a receding horizon heuristic similar to that found in [4] in which the best solution was chosen from solving several MILP problems defined over smaller subdivided regions of the original region.

6 Traffic detection algorithms

We now describe two very distinctive traffic change detection algorithms. The first algorithm identifies traffic changes by using inflection points of a cubic b-spline representation of the data. The second algorithm uses the Hellinger distance to determine when the traffic distribution changes. These two different approaches, when combined with our adaptive scheduling and placement algorithms, demonstrates the consistency of our adaptive approach as discussed in the next section.

6.1 Inflection point detection algorithm

Suppose that sensors $s_j, \ldots, s_k$ are actively detecting traffic moving through B. However, if the traffic distribution $\mathcal{A}$ shifts to the left or right, we want to detect this shift. Our approach assumes that if the traffic continuously moves away from active sensors, the total number of detections over time should decrease. This corresponds to a change in concavity in the number of detections over time that results in a loss of detections if no other sensors are activated. Our method of finding this change in concavity is to smooth potentially noisy data and numerically determine inflection points that satisfy the condition as described. Figure 3 demonstrates this idea conceptually.

Let the number of detections from all sensors be $N_d = \{d_1, \ldots d_N\}$. We approximate N_d as a smooth function $\mathcal{D}(t)$ that captures the major trends in the data by using b-splines [9] as

$$\mathcal{D}(t) = \sum_{i=-1}^{N+2} d_i' \mathcal{B}_i^3(t) \tag{36}$$

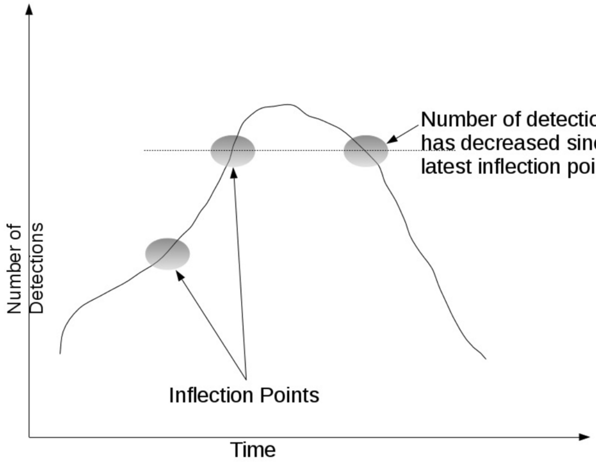
Number of detections has decreased since latest inflection point
Number of Detections
Inflection Points
Time

Fig. 3 Detecting a shift in the traffic distribution $\mathcal{A}$ by detecting the drop in number of detections

where $\mathcal{B}_i^3(t)$ is defined over the knot sequence defined as $N_k = \frac{k}{N+1}$ for $-3 \le k \le N+4$. We impose the conditions $d'_{-1} = d'_0 = d'_1 = d_1, d'_N = d'_{N+1} = d'_{N+2} = d_N$, to clamp the spline at the endpoints, which has the effect of interpolating the data at the endpoints so that $\mathcal{D}(0) = d_1$ and $\mathcal{D}(1) = d_N$. Note that our b-spline representation does not intentionally interpolate the data anywhere else, as it uses the detection data over time as the control points of the associated cubic spline.

Let $S = \{\mu : \mathcal{D}''(\mu) = 0\}$ be the set of inflection points and let

$$p^* = \max_{p \in S} \mathcal{D}(p). \tag{37}$$

If there exists q^* such that $q^* \ne p^*$ and $\mathcal{D}(p^*) = \mathcal{D}(q^*)$, then we identify this event as a change in traffic.

6.2 Hellinger detection algorithm

The Hellinger detection algorithm constructs a distribution of detections over the sensors, which we denote as h. At the k−th time step, the new detections, denoted as $\delta^{(k)}$, are aggregated with h to form the new distribution $h^{(k)}$. We can compare these two distributions by computing the Hellinger distance, defined as

$$d^2\left(h, h^{(k)}\right) = \frac{1}{2} \sum_{i=1}^{M} \left(\sqrt{h_i} - \sqrt{h_i^{(k)}} \right)^2, \tag{38}$$

which is a special, normalized case of the α−order Hellinger distance considered in [5]. If $d(h, h^{(k)}) > \alpha$ for some threshold $\alpha > 0$, then the distributions are sufficiently different, $h = \delta^{(k)}$, and we identify this event as a traffic change event. Otherwise, no change is detected and $h = h^{(k)}$.

7 Simulation results

We compare four distinctive traffic monitoring solutions. The first, which we refer to as PF (Potential Field), uses the potential field based scheduling algorithm, the MILP placement approach for heterogeneous environments, and the inflection point detection algorithm. The second method, which we refer to as PFH (Potential Field Hellinger) uses almost all of the same algorithms as PF, except that it uses the Hellinger based detection algorithm. The third method is MMC (Maximum Minimum Coverage), which is an area coverage algorithm that discretizes the field into grid points that are covered by sensors so as to minimize the probability of missing detections at each grid point [11]. The final method is AA (All Active), which is a path coverage method in which all sensors are active based upon the method found in [20]. All sensors are placed in an active state, since the arrival distribution is unknown. The AA method uses the same sensor positions as PFH and PH; keeping the same positions allows for comparison of the different scheduling strategies. Sample output of both our placement algorithm and that of MMC can be seen in Figs. 4 and 5, respectively.

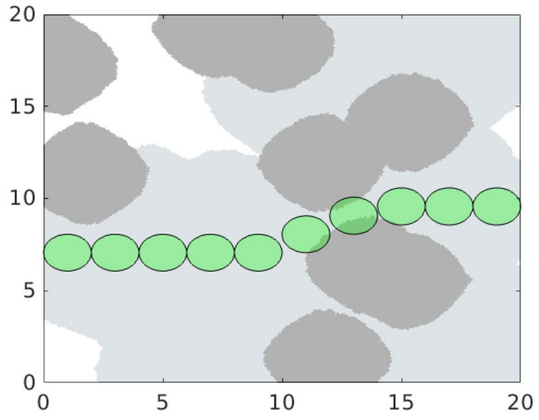

Fig. 4 Sensor positions using the sensor placement algorithm with sensor radius $R = 1$ as described in Sect. 5. The green circles represent sensor regions

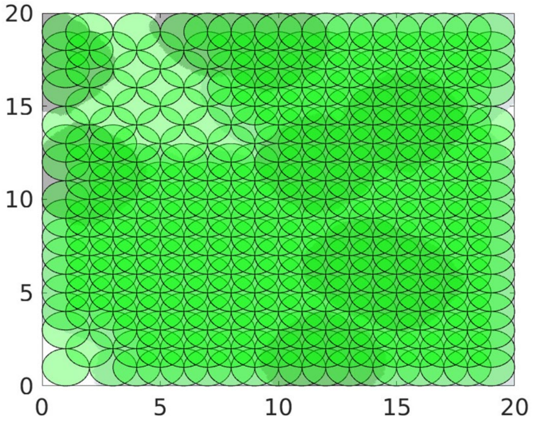

Fig. 5 Maximum minimum coverage of an area with sensor radius $R = 1$ using 334 sensors

For each run of the simulation, we sample traffic from the distribution

$$(1 - \upsilon)U[x, x+2] + \upsilon U[0, 20] \tag{39}$$

where $0 \le \upsilon \le 1$ is a homotopy parameter that regulates the influence of a temporally adjustable distribution $U[x, x+2]$ with constant speed of 1 unit of distance per unit of time and a static uniform distribution $U[0, 20]$. We chose these two distributions to see how well our scheduling approach works with a known mobile distribution, as well as how it is affected as the underlying traffic distribution deforms to a completely static distribution defined for the entire boundary β. For each homotopy value, we make 30 simulation runs in which all four methods use the same traffic distribution, as well as the same traffic samples. To compare all four approaches, we estimate the probability of detections, as well as efficiency, which we define as

$$e = \frac{N_d}{N_t N_a}, \tag{40}$$

where N_d is the total number of detected ships, N_t is the total number of ships, and N_a is the total number of active time units for all sensors.

Figure 6 shows the probability of detected ships for all four methods. For PFH, we set $\alpha = .5$. Not surprisingly, the area coverage approach had the highest probability of detected ships. The AA path coverage method does quite well with regard to the other path coverage methods that we compare as determined by the probability of detections. We also see that both the PF and PFH methods are comparable with respect to the probability of detections. As $\upsilon \rightarrow 1$, we see a small decrease in the performance of both the PF and PFH methods, which we attribute to an increasing probability of traffic appearing outside of $U[x, x+2]$. The same trend is also seen in Fig. 7 which shows the efficiency of all methods. However, both PF and PFH are clearly more efficient than MMC, which is caused by the large number of sensors used by MMC. Both methods are also more efficient than AA, since all of the sensors using the AA method waste more time actively waiting for detections that do not occur.

Figure 8 shows no significant difference for the PF and PFH method. However, Fig. 9 shows that if we decrease α to .1, we see a change in performance; the PFH method performs more efficiently than the PF method. As α controls the sensitivity of the Hellinger-based detector algorithm, it is tempting to always set the sensitivity to something small. If the sensor placement algorithm is developed to favor sensor performance, as is the case with our sensor placement algorithm, then small α values are desired. However, in the case where another sensor placement algorithm chooses sensor positions that yield a large variance in the quality of sensor performance, smaller α values may be problematic for the following reason. If α is chosen where any change in the distribution results in a traffic change detection, then the scheduling algorithm could unnecessarily choose to activate a sensor of

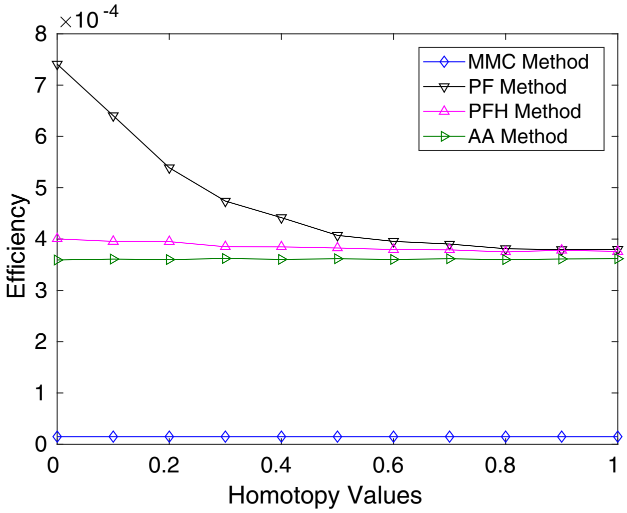

Fig. 7 Efficiency, $\alpha = .5$ for the Hellinger-based traffic change detection method

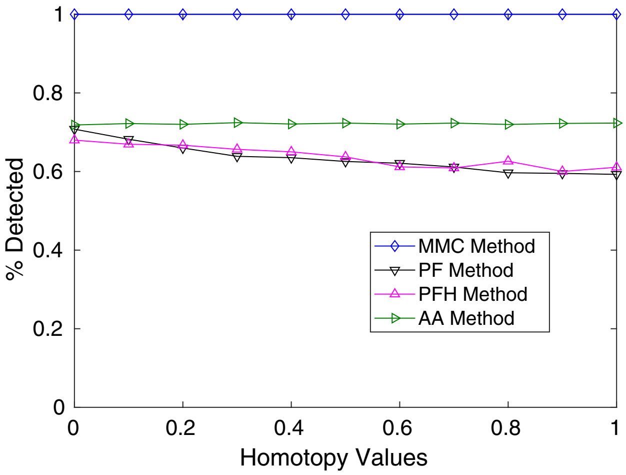

Fig. 6 Probability of detection, with $\alpha = .5$ for the Hellinger-based traffic change detection method

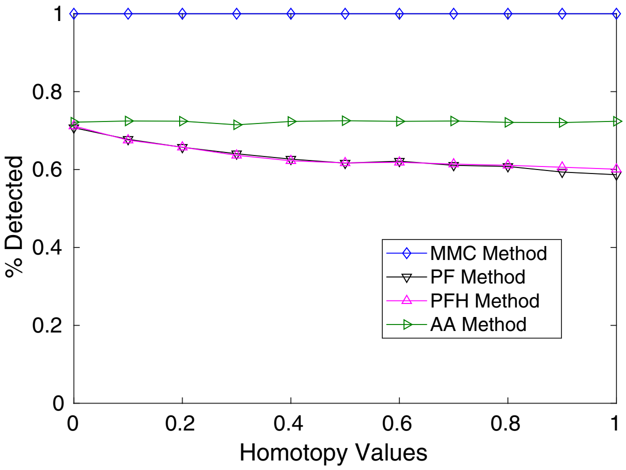

Fig. 8 Probability of detection, $\alpha = .1$ for the Hellinger-based traffic change detection method

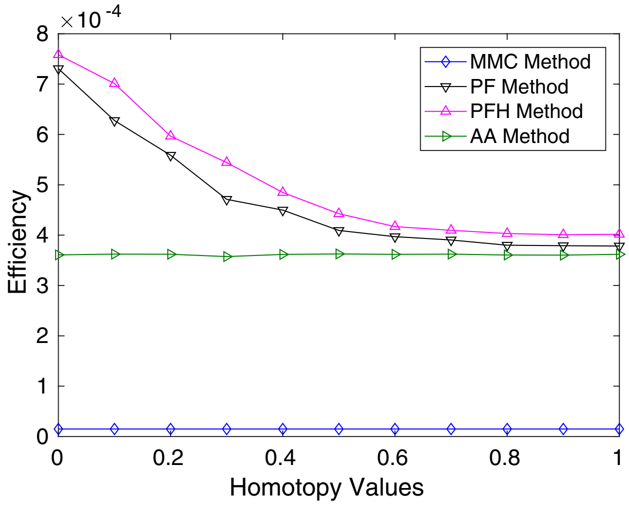

Fig. 9 Efficiency, $\alpha = .1$for the Hellinger-based traffic change detection method

a poorer quality than the currently active sensor. This is a conjecture whose validation is the subject of future work. Nonetheless, the higher efficiency results of both PF and PFH methods as compared to both MMC and AA methods demonstrate the consistent efficiency of our adaptive scheduling algorithm.

8 Summary

We have provided a potential field based algorithm that works very well with regard to efficient monitoring of traffic. This algorithm is general and requires a sensor placement algorithm and a traffic detection algorithm. To demonstrate the utility of this algorithm, we created a sensor placement algorithm, two traffic change detection algorithms, and combined these algorithms to create two distinctive traffic monitoring solutions that use the same placement algorithm. We compared our traffic monitoring solutions to an area coverage method and all-active path coverage method using a homotopy method to vary the influence of a static and a mobile distribution. Our results demonstrated the efficiency of our approach, which we believe has wide application to both sea and land domains.

Author contributions RDT, JCH, and MJB originally developed and matured the original concept of adaptive scheduling. RDT and BH developed and discussed proofs of the main results. RDT implemented the software to test the algorithms and drafted the manuscript. All authors contributed to numerous edits and versions of this draft.

Funding This work is funded by the Office of Naval Research (ONR), Code 32, and the Independent Applied Research (IAR) program at NSWC PCD.

Compliance with ethical standards

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

Consent for publication Not applicable.

Ethics approval and consent to participate Not applicable.

References

1. Akyildiz I, Pompili D, Melodia T (2005) Underwater acoustic sensor networks: research challenges. Ad Hoc Netw J 3(3):257–279
2. Åkesson S (2002) Tracking fish movements in the ocean. Trends Ecol Evol 17:56–57
3. Barraquand J, Langlois B, Latombe JC (1992) Numerical potential field techniques for robot path planning. IEEE Trans Syst Man Cybern 22(2):224–241
4. Bellingham J, Richards A, How JP (2002) Receding horizon control of autonomous aerial vehicles. In: Proceedings of the American control conference, vol 5, pp 3741–3746. IEEE
5. Burbea J, Rao CR (1982) Entropy differential metric, distance and divergence measures in probability spaces: a unified approach. J Multivar Anal 12(4):575–596
6. Cardei M, MacCallum D, Cheng MX, Min M, Jia X, Li D, Du DZ (2002) Wireless sensor networks with energy efficient organization. J Interconnect Netw 3(03n04):213–229
7. Cardei M, Thai MT, Li Y, Wu W (2005) Energy-efficient target coverage in wireless sensor networks. In: Proceedings IEEE INFOCOM 2005, 24th annual joint conference of the IEEE computer and communications societies, vol 3, pp 1976–1984. IEEE
8. Chicone C (2006) Ordinary differential equations with applications. Springer, Berlin
9. de Boor C (1978) A practical guide to splines. Springer, Berlin
10. Demirbas M, Ferhatosmanoglu H (2003) Peer-to-peer spatial queries in sensor networks. In: Proceedings third international conference on peer-to-peer computing, 2003 (P2P 2003), pp 32–39. IEEE
11. Dhillon SS, Chakrabarty K (2003) Sensor placement for effective coverage and surveillance in distributed sensor networks. In: 2003 IEEE wireless communications and networking. WCNC 2003, vol 3, pp 1609–1614. IEEE
12. Fuemmeler JA, Veeravalli VV (2008) Smart sleeping policies for energy-efficient tracking in sensor networks. In: Networked sensing information and control, pp 267–287. Springer, Berlin
13. GonzáLez-Castro V, Alaiz-RodríGuez R, Alegre E (2013) Class distribution estimation based on the Hellinger distance. Inf Sci 218:146–164
14. Hartman BT, Tatum RD, Bays MJ (2018) Heterogeneous sensor-robot team positioning and mixed strategy scheduling. In: 2018 IEEE/RSJ international conference on intelligent robots and systems (IROS), pp 3137–3144. IEEE
15. Hwang Y, Ahuja N (1992) A potential field approach to path planning. IEEE Trans Robot Autom 8(1):23–32
16. Hyland J, Tatum R, Hatcher J (2017) Analysis of harbor protection systems with strongly correlated sensors. In: OCEANS–Anchorage, 2017, pp 1–4. IEEE

17. McIntyre GA, Hintz KJ (1996) Information theoretic approach to sensor scheduling. In: Signal processing, sensor fusion, and target recognition V, vol 2755, pp 304–313. International Society for Optics and Photonics
18. Shipps JC, Abraham BM (2004) The use of vector sensors for underwater port and waterway security. In: Proceedings the ISA/IEEE sensors for industry conference, 2004, pp 41–44. https://doi.org/10.1109/SFICON.2004.1287125
19. Slijepcevic S, Potkonjak M (2001) Power efficient organization of wireless sensor networks. In: ICC 2001 IEEE international conference on communications, vol 2, pp 472–476. IEEE
20. Tatum R, Bays M (2016) Decentralized placement of heterogeneous nodes. In: OCEANS 2016 MTS/IEEE Monterey, pp 1–6
21. Unser M, Aldroubi A, Eden M (1993) B-spline signal processing. II. Efficiency design and applications. IEEE Trans Signal Process 41(2):834–848
22. Wu D, Chatzigeorgiou D, Youcef-Toumi K, Mekid S, Ben-Mansour R (2014) Channel-aware relay node placement in wireless sensor networks for pipeline inspection. IEEE Trans Wirel Commun 13(7):3510–3523
23. Yick J, Mukherjee B, Ghosal D (2005) Analysis of a prediction-based mobility adaptive tracking algorithm. In: 2nd international conference on broadband networks, 2005. BroadNets 2005, pp 753–760. IEEE

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.