Multi-Objective Continuous Sensor Scheduling for Long Horizon Path Planning

Christopher Gilliam*, Bisma Amjad*, Marek Hilton*, Sam Pike†, Jordi Barr†, Alex Kenyon†, Nicola Perree†, Marco Martorella*, Bill Moran‡ and Beth Jelfs*
*School of Engineering, University of Birmingham, Birmingham, UK
{c.gilliam.1, b.amjad, m.y.y.hilton, m.martorella, b.jelfs}@bham.ac.uk
†Cyber and Information Systems, Defence Science and Technology Laboratory, Salisbury, UK
{spike, jmbarr, akenyon, nperree}@dstl.gov.uk
‡Dept. Electrical & Electronic Engineering, University of Melbourne, Melbourne, Australia
wmoran@unimelb.edu.au

***Abstract*—Optimal planning for autonomous vehicles over long time horizons, where sensor information and multiple objectives form part of the decision-making process, is still in its infancy. The challenges of uncertainty, inherent in sensor information, and the computational complexity of their implementation typically render direct approaches infeasible. We present a practical, close to optimal, approach to the problem of long horizon path planning and goal-seeking for an autonomous mobile platform with sensing capability. Optimal autonomous vehicle planning is notoriously difficult, presenting several significant challenges: handling the uncertainty inherent in the sensor data and in a potentially unknown environment; coping with the ostensibly large data storage requirements; and the computational complexity of looking many epochs ahead. To overcome these problems, our model-based solution leverages a stochastic search methodology to obtain long-term, continuous, trajectories. We demonstrate its capability in handling both uncertainty in sensor measurements as well as multiple, possibly conflicting, objectives.**

***Index Terms*—sensor scheduling, path planning, trajectory optimization, RRT*.**

I. Introduction

Autonomous mobile platforms have begun to be seen as key technologies for terrestrial, maritime and air systems [1]–[3]. Given a specified goal, true autonomy requires the platform to interact with complex environments and take actions, without human intervention, towards achieving the goal. A canonical example is one where an autonomous vehicle with several sensors, such as radar, lidar and infra-red cameras or, in a maritime environment, sonars and magnetometers, seeks to navigate a complicated terrain while attempting to locate and track a dynamic target. Planning for autonomous mobile platforms over long time horizons, where sensor information forms part of the trajectory decision making process, and where the platform has multiple, possibly conflicting objectives, has only recently begun to be addressed [4]–[8].

These problems can be formulated in terms of an underlying state-space, an action/decision space, and a set of measurements that provide information about the current state. Such problems are often expressed in terms of Partially Observed Markov Decision Processes (POMDPs) [9], and are notoriously difficult to solve because of the high computational complexity associated with their solutions. Typically, simplifications of the structure are used to reduce memory requirements and computation time, e.g., constraining the action space [1] or policy rollout [10]. Even so, attempts to solve these simplified versions, that go beyond a single-step greedy approach, face an exponentially growing range of actions as the planning horizon extends. Additionally, planning algorithms need to mitigate the uncertainty in sensor measurements and target objectives; this uncertainty can propagate and influence future actions. The combination of uncertainty in both measurements and target objectives differentiates sensor scheduling problems from standard stochastic planning problems. Accordingly, there is a need for practical long horizon sensor scheduling algorithms that are resilient and computationally tractable.

The trajectory planning approach used here adapts concepts prevalent in the robotics community; specifically, the Rapidly exploring Random Tree Star (RRT*) algorithm [11]. The core idea is to obtain trajectories by *randomly* propagating a tree of future states, and modelling the uncertainty in future measurements using known sensor characteristics. The structure of the RRT* algorithm yields a computational complexity which is bounded, not by the size of the action space, but by the cost of inserting a new node in the tree. The method used in the RRT* algorithm for node insertion (rewiring) offers probabilistic guarantees on performance. As a result, the RRT* algorithm is often used in standard path planning problems [2], [3], [12], i.e., travelling from A to B whilst avoiding obstacles. In our previous work [5], [6], we adapted the RRT* for trajectory planning while minimizing not only motion costs but also an information theoretic cost relating to sensor performance.

In this paper, we now consider long-horizon path planning for a mobile sensing platform faced with multiple, possibly competing, objectives; searching for unknown targets whilst maintaining tracking of known targets. Building upon our earlier methodology [13]–[15], this multi-objective problem is handled via a Pareto convex combination of the individual cost functions. The planner's search cost is captured using a search map; a grid-based representation of the operational

This work was supported by the UK MOD through the Defence and Security Accelerator's "*Autonomous Sensor Management and Sensor Counter Deception*" competition.

area where each grid cell specifies the probability that an undetected target exists in that cell [16], [17]. The tracking cost is captured using a standard information theoretic cost based on tracking [18], [19]. The Pareto cost is then used in our modified RRT* algorithm [5], [6] to generate the sensor's path. Using this framework, we validate our approach for a track-and-search scenario for an autonomous mobile platform in a structured environment and provide a characterization of the problem by adjusting the Pareto weight.

II. Mathematical Models

A. Platform & Sensor Models

The platform at time k is represented by the state $\boldsymbol{x}_k^p = [x_k^p, v_{x,k}^p, y_k^p, v_{y,k}^p]^\mathrm{T}$, where $\boldsymbol{p}_k^p = [x_k^p, y_k^p]^\mathrm{T}$ is the Cartesian position of the platform and $\boldsymbol{v}_k^p = [v_{x,k}^p, v_{y,k}^p]^\mathrm{T}$ is its velocity vector. The heading ψ_k^p of the platform at time k is

$$\psi_k^p = \arctan\left(\frac{v_{y,k}^p}{v_{x,k}^p}\right) \in (-\pi, \pi]. \tag{1}$$

The platform has an active sensor measuring range, r, and azimuth, θ, with the sensor frame aligned with the body frame of the platform. Measurements are modelled as the true measurement with added random Gaussian noise. Accordingly, the vector of measurements at time k is:

$$\boldsymbol{z}_k = h\left(\boldsymbol{x}_k^t, \boldsymbol{x}_k^p\right) + \boldsymbol{w}_k, \tag{2}$$

where $\boldsymbol{x}_k^t$ is the target state, $\boldsymbol{w}_k$ is zero-mean Gaussian noise with covariance $\boldsymbol{R}$, and $h(\boldsymbol{x}_k^t, \boldsymbol{x}_k^p)$ represents the measurement process:

$$\text{Range:}\quad h_r(\boldsymbol{x}_k^t, \boldsymbol{x}_k^p) = \left|\boldsymbol{p}_k^t - \boldsymbol{p}_k^p\right| \tag{3}$$

$$\text{Azimuth:}\quad h_\theta(\boldsymbol{x}_k^t, \boldsymbol{x}_k^p) = \arctan\left(\frac{y_k^p - y_k^t}{x_k^p - x_k^t}\right) - \psi_k^p. \tag{4}$$

B. Target State

The target at time k is represented by the state $\boldsymbol{x}_k^t = [x_k^t, v_{x,k}^t, y_k^t, v_{y,k}^t]^\mathrm{T}$, where $\boldsymbol{p}_k^t = [x_k^t, y_k^t]^\mathrm{T}$ is the Cartesian position of the platform and $\boldsymbol{v}_k^t = [v_{x,k}^t, v_{y,k}^t]^\mathrm{T}$ is its velocity vector. We adopt a standard constant velocity model for target motion, perturbed by a continuous white noise acceleration [20]: the transition of a target state from time k to time $k+1$ is:

$$\boldsymbol{x}_{k+1}^t = \boldsymbol{f}(\boldsymbol{x}_k^t) + \boldsymbol{e}_k = \left(\begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \otimes \boldsymbol{I}_2\right) \boldsymbol{x}_k^t + \boldsymbol{\eta}_k, \tag{5}$$

where T represents the time between step k and $k+1$, $\otimes$ is the Kronecker product and $\boldsymbol{I}_2$ is the 2×2 identity matrix. The vector $\boldsymbol{\eta}_k$ represents the process noise: a zero-mean Gaussian variable with a typical covariance $\boldsymbol{Q}$:

$$\boldsymbol{Q} = q\left(\begin{bmatrix} \frac{T^3}{3} & \frac{T^2}{2} \\ \frac{T^2}{2} & T \end{bmatrix} \otimes \boldsymbol{I}_2\right), \tag{6}$$

where q represents the magnitude of the process noise; expressing the deviation of the target motion from the constant velocity model. The scheduler is assumed to have prior knowledge of the motion model and process noise covariance, i.e., there is no model mismatch.

III. Multi-Objective Scheduling Framework

The current information the planner has regarding the problem at a given point in time is described by the *belief state* and includes the following:

- Target states – estimated states of target(s) being tracked;
- Track uncertainty – estimated covariance matrices associated with the target states;
- Search map – current knowledge of undetected targets in the search area represented by the search map.

The belief state update is driven by the search and tracking objectives; the tracker uses new measurements to update target state information and the search map is updated according to the movement of the platform.

For a given action, the cost function is constructed as a Pareto sum of the multiple objectives the platform is attempting to achieve. In our case, the cost function for an action $\boldsymbol{a}$ is

$$\mathcal{J}(\boldsymbol{a}) = \beta \mathcal{J}_{track}(\boldsymbol{a}) + (1-\beta)\mathcal{J}_{search}(\boldsymbol{a}), \tag{7}$$

where $\mathcal{J}$ is the combined cost function, β is the weighting of the combination and $\mathcal{J}_{track}$ and $\mathcal{J}_{search}$ are the individual costs, respectively, for the tracking and search objectives, as discussed in our previous work [13]–[15].

A. Tracking Cost

Due to the nonlinear nature of the measurements we opt for an extended Kalman filter (EKF) for the tracking algorithm. The Jacobians for the measurement and target transition functions are:

$$\boldsymbol{H}_k(\hat{\boldsymbol{x}}_{k+1|k}^t, \boldsymbol{x}_k^p) = \nabla \boldsymbol{h}(\boldsymbol{x}_k^t, \boldsymbol{x}_k^p)\big|_{\boldsymbol{x}_k^t = \hat{\boldsymbol{x}}_{k+1|k}^t} \tag{8}$$

$$\boldsymbol{F} = \nabla \boldsymbol{f}(\boldsymbol{x}_k^t)\big|_{\boldsymbol{x}_k^t = \hat{\boldsymbol{x}}_{k+1|k}^t}, \tag{9}$$

where ∇ represents differentiation with respect to the state $\boldsymbol{x}_k^t$.

During the time interval T, the platform motion is controlled by a velocity V_k and angular direction ϕ_k, so that the next platform state is

$$\begin{bmatrix} x_{k+1}^p \\ v_{x,k+1}^p \\ y_{k+1}^p \\ v_{y,k+1}^p \end{bmatrix} = \begin{bmatrix} x_k^p + V_k T\cos(\phi_k) \\ V_k \cos(\phi_k) \\ y_k^p + V_k T \sin(\phi_k) \\ V_k \sin(\phi_k) \end{bmatrix}. \tag{10}$$

An action vector, $\boldsymbol{a} = [V, \phi]^\mathrm{T}$, represents the motion control of the platform. The cost of an action or action sequence is determined based on the Posterior Cramér Rao bound (PCRB) [18], [19]. The PCRB is the predicted covariance matrix for a target, based on future target movement and sensor measurements. Using previous trajectory information, the platform can predict where the target will move next. This predicted movement can then be used to predict informational content of sensor measurements from the possible actions; the action that results in the highest informational content will reduce the uncertainty in tracking the target. In the following, we give the expression for the PCRB for our scenario however for a full derivation we refer the reader to [18] and [21, Ch.4].

For an action $\boldsymbol{a}$, the PCRB has two parts:

$$\tilde{\boldsymbol{P}}(\boldsymbol{a}) = \left[\boldsymbol{J}_p + \boldsymbol{J}_m(\boldsymbol{a})\right]^{-1}, \tag{11}$$

where $\boldsymbol{J}_p$ is the prior information matrix, and $\boldsymbol{J}_m(\boldsymbol{a})$ is the predicted measurement information matrix, which describes the information gained from measuring the target after applying action $\boldsymbol{a}$. These information matrices are:

$$\boldsymbol{J}_p = \left[\boldsymbol{F}\boldsymbol{P}_k\boldsymbol{F}^{\mathrm{T}} + \boldsymbol{Q}\right]^{-1} \tag{12}$$

$$\boldsymbol{J}_m(\boldsymbol{a}) = P_d\, \boldsymbol{H}_k^{\mathrm{T}}(\boldsymbol{a})\boldsymbol{R}^{-1}\boldsymbol{H}_k(\boldsymbol{a}), \tag{13}$$

where P_d is the probability of detecting the target, and $\boldsymbol{H}_k(\boldsymbol{a}) \triangleq \boldsymbol{H}_k(\hat{\boldsymbol{x}}^t_{k+1|k}, \boldsymbol{x}^p_k, \boldsymbol{a})$, subject to moving the platform using action $\boldsymbol{a}$. For an action sequence, the PCRB computation is recursive, with the previous PCRB matrix taking the place of $\boldsymbol{P}_k$ beyond the first step. The EKF tracker provides a closed form for $\boldsymbol{J}_m$.

The cost function, based on the average variance of the target state after taking action $\boldsymbol{a}$, is

$$\mathcal{J}_{\text{track}}(\boldsymbol{a}) = \text{tr}\left(\tilde{\boldsymbol{P}}(\boldsymbol{a})\right) \tag{14}$$

where $\text{tr}(\cdot)$ is the trace of the matrix.

B. Search Cost

Control of searching for unknown targets in the scene uses the concept of a search map (see [16], as well as [13]–[15], [17], [22]) to store information relating to undetected targets in the operational area. The (m, n) cell, with central coordinates (x_m, y_n), in the search map is assigned the probability $P_T(x_m, y_n)$, and represents the probability of an undetected target in that cell. Note that, in general, this is not a probability density/distribution on the set of cells, however, it amounts to that if it is known that there is only one target. Initially, when no information is known, the probability of undetected targets for all cells in the search map is relatively high. These probabilities are changed by the platform's sensor probing the operational area and resolving whether targets are present. The search map guides the scheduling of future measurements – increasing focus on areas where the search cost is higher (within the constraints of the sensors and platform motion). This continues until the area is cleared.

The update of the search map probabilities at epoch k resulting from a sensor measurement is calculated by applying a Bayesian update to each cell of the map, thus:

$$P_{T,k}(x_m, y_n) = \frac{(1 - P_d(x_m, y_n))P_{T,k-1}(x_m, y_n)}{(1-P_d(x_m, y_n))P_{T,k-1}(x_m, y_n) + (1-P_f)(1-P_{T,k-1}(x_m, y_n))} \tag{15}$$

where, for a given sensor, P_d and P_f are respectively the probability of detection and of a false alarm. Note that P_d is defined per cell allowing for the incorporation of realistic detection mechanisms. Also, in this work, the false alarm rate is assumed to be very low so we do not consider an update for a detection that could be a false alarm.

Given this update mechanism, the search cost function is based on the reduction in total probability across the search map. Thus, assuming $P_{T,k,\boldsymbol{a}}(x_m, y_n)$ is the predicted threat map at time k for action $\boldsymbol{a}$, then the search cost is

$$\mathcal{J}_{\text{search}}(\boldsymbol{a}) = \sum_{x_m}\sum_{y_n} P_{T,k,\boldsymbol{a}}(x_m, y_n) \tag{16}$$

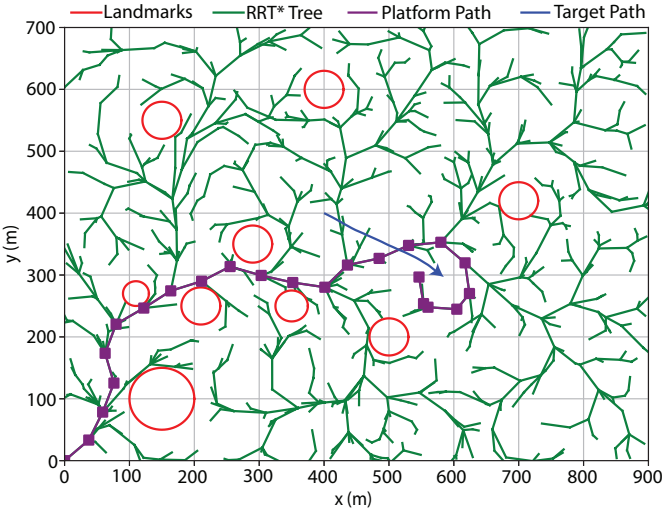
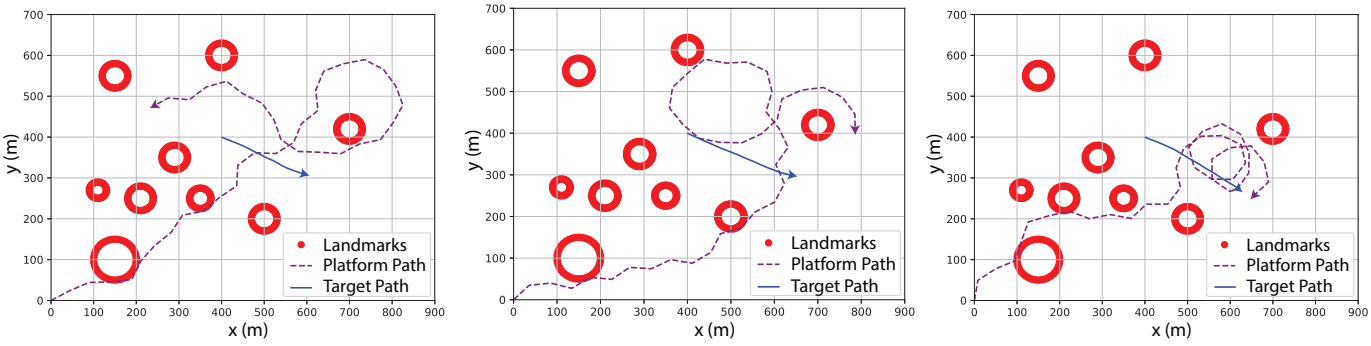


Fig. 1. Example of a RRT* tree and the resulting platform path. The tree is constructed using $N_s = 1000$ samples and considering only the tracking cost. The platform's path is shown in purple with the purple squares indicating different time epochs. The target's trajectory is marked in blue with an arrow indicating direction of travel.

C. Rapidly Exploring Random Trees (RRT*)

The RRT* algorithm is used to produce long-horizon paths based on the current belief state and position of the platform. The algorithm functions as follows: First, a random point is chosen in the search area. Next, a virtual move is made from the nearest vertex on the existing tree in the direction of the random point (given the physical constraints of the environment) to create a new node. This new node is initially connected to the nearest vertex. Third, this initial connection is refined in a rewiring step. This step involves computing the cost – using (7) – of reaching the new node from existing vertices within a local neighbourhood of the new node. The new node is rewired to the vertex that corresponds to the minimum cost. These steps are repeated for N_s random samples to construct the tree. Once constructed, the platform's path corresponds to the path from the tree's root to the vertex with the lowest cost. An example of an RRT* tree and the resulting path are shown in Figure 1. In this example, the horizon is 22 epochs in length, but this will vary depending on the information available to the scheduler and the value of β in the cost function.

In the traditional algorithm [11], the rewiring process enables the RRT* to provide asymptotic optimality. However, when the goal state is flexible or unknown, such as approaching a moving target, the output from the RRT* is only optimal in that time instance [23]. For example, the path determined from the RRT* may not be optimal, in terms of the cost (7), if a new target is found. Accordingly, to take advantage of



(a) Path with $\beta = 0$ (b) Path with $\beta = 0.4$ (c) Path with $\beta = 1$

Fig. 2. Example platform paths generated by the scheduling algorithm. Parts (a), (b) and (c) respectively show the paths generated when $\beta = 0$, 0.4 and 1. In each figure, the obstacles are shown in red, target trajectory in dashed blue and the platform path in dashed purple. Arrows indicate direction of travel.

new information obtained by the platform, the RRT* is re-run every time epoch and only the first action is taken.

The usefulness of the RRT* lies in the tree structure, which offers two key advantages: reduced computation time compared to other methods, and simplification in handling modelling errors. Random sampling of the spatial domain to construct the tree from which the path is developed, massively reduces the computational complexity to only the insertion cost of a new node in the tree; for N_s samples, this cost is $\mathcal{O}(N_s \log(N_s))$ [11]. A specific benefit of RRT* is that only a local search, within a specified distance d (see [23]), is necessary in the exploration stage.

In this report, we use knowledge of the search map and target tracks to guide the random sampling in the RRT*. When tracking, samples are drawn from a Gaussian distribution centered on the target's estimated location. In contrast, when searching, samples are drawn from the unexplored, high probability, areas of the search map. The value of β, see (7), dictates the proportion of samples drawn from these two distributions.

IV. Simulations

We now present simulation results for our multi-objective continuous sensor scheduler. The scheduler is tasked with planning a platform's path to both search an area and track any targets it finds. The scenario for the simulations is shown in Figure 2; the environment contains obstacles that are impassable and occlude sensing, and a single target. Planning performance is assessed based on tracking accuracy and we compare the performance as the weight β varies. Tracking accuracy is measured using the Optimal Sub-Pattern Assignment (OSPA) [24] metric, that is, the Euclidean distance between the track estimate and target's true state with a maximum distance of association to 100 m. Finally, the number of samples in the RRT* is set to $N_s = 1000$. Note that all simulations (including the tracker) are implemented in Stone Soup [25]–[27].

Examples of the paths generated by the scheduler are shown in Figure 2 and median OSPA track error is shown in Figure 3 for $\beta = 0, 0.2, 0.4, 0.6, 0.8, 1$. The figures demonstrate the scheduler's ability to plan the trajectory of the platform in a

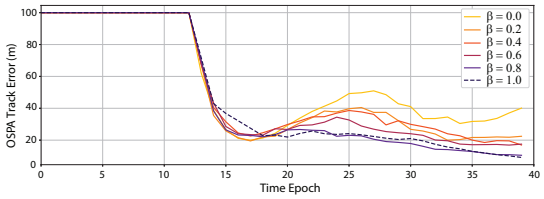


Fig. 3. Graph showing the median OSPA tracking error for the platform for different values of β. Results are averaged over 200 Monte-Carlo simulations.

structured environment to find and track the target accurately. Initially the platform searches until it finds the target then it's behaviour depends on the choice of β. For example, if $\beta = 1$ then the platform circles the target to maintain tracking performance, see Figure 2c. In contrast, if $\beta = 0$ the platform carries on searching and does not prioritize tracking, see Figure 2a. This change in behaviour is supported by the tend in tracking performance. Figure 3 demonstrates that the median tracking performance improves as β increases, which is expected as the scheduler is weighted more to the tracking objective.

V. Conclusion

We presented a multi-objective framework for the long-horizon scheduling of an autonomous moving platform tasked with exploring an area and tracking any targets it finds. Our framework uses a stochastic RRT* algorithm to generate long-term, continuous, trajectories for the platform. The competing objectives – search and track – are combined within the RRT* as a Pareto sum of a tracking cost and a search cost; an adjustable weight parameter allows for balancing of the costs. Using simulations, we evaluated the performance of our scheduling framework for a structured environment with a single unknown target.

VI. Acknowledgment

The simulations described in this paper were performed using the University of Birmingham's BlueBEAR High Performance Computing service. See http://www.birmingham.ac.uk/bear for more details.

REFERENCES

[1] P. Braca, R. Goldhahn, K. D. LePage, S. Marano, V. Matta, and P. Willett, "Cognitive multistatic AUV networks," in *17th International Conference on Information Fusion (FUSION)*, Salamanca, Spain, July 2014, pp. 1–7.

[2] R. Zaccone, "COLREG-compliant optimal path planning for real-time guidance and control of autonomous ships," *Journal of Marine Science and Engineering*, vol. 9, no. 4, p. 405, 2021.

[3] A. Lazarowska, "Review of collision avoidance and path planning methods for ships utilizing radar remote sensing," *Remote Sensing*, vol. 13, no. 16, p. 3265, 2021.

[4] D. Angley, B. Ristic, W. Moran, and B. Himed, "Search for targets in a risky environment using multi-objective optimisation," *IET Radar, Sonar & Navigation*, vol. 13, no. 1, pp. 123–127, 2019.

[5] X. Wang, S. Williams, D. Angley, C. Gilliam, T. Jackson, R. Ellem, A. Bessell, and B. Moran, "RRT* trajectory scheduling using angles-only measurements for AUV recovery," in *22nd International Conference on Information Fusion (FUSION)*, Ottawa, ON, Canada, Jul. 2019, pp. 1–6.

[6] S. Williams, X. Wang, D. Angley, C. Gilliam, B. Moran, R. Ellem, T. Jackson, and A. Bessell, "Dynamic target driven trajectory planning using RRT," in *22nd International Conference on Information Fusion (FUSION)*, Ottawa, ON, Canada, Jul. 2019, pp. 1–8.

[7] F. Hoffmann, A. Charlish, M. Ritchie, and H. Griffiths, "Sensor path planning using reinforcement learning," in *2020 IEEE 23rd International Conference on Information Fusion (FUSION)*, July 2020, pp. 1–8.

[8] ——, "Policy rollout action selection in continuous domains for sensor path planning," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 57, no. 4, pp. 2247–2264, Aug 2021.

[9] A. Hero and D. Cochran, "Sensor management: Past, present, and future," *IEEE Sensors Journal*, vol. 11, no. 12, pp. 3064–3075, Dec 2011.

[10] D. P. Bertsekas and D. A. Castanon, "Rollout algorithms for stochastic scheduling problems," *Journal of Heuristics*, vol. 5, no. 1, pp. 89–108, 1999.

[11] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011.

[12] M. Elbanhawi and M. Simic, "Sampling-based robot motion planning: A review," *IEEE Access*, vol. 2, pp. 56–77, 2014.

[13] D. Angley, S. Mehrkanoon, B. Moran, C. Gilliam, and S. Simakov, "Improving automated search for underwater threats using multistatic sensor fields by incorporating unconfirmed track information," in *IEEE International Conference on Autonomous Systems (ICAS)*, Montreal, QC, Canada, Aug. 2021, pp. 1–5.

[14] C. Gilliam, B. Ristic, D. Angley, S. Suvorova, B. Moran, F. Fletcher, H. Gaetjens, and S. Simakov, "Scheduling of multistatic sonobuoy fields using multi-objective optimization," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Calgary, AB, Canada, Apr. 2018, pp. 3206–3210.

[15] C. Gilliam, D. Angley, S. Williams, B. Ristic, B. Moran, F. Fletcher, and S. Simakov, "Covariance cost functions for scheduling multistatic sonobuoy fields," in *21st International Conference on Information Fusion (FUSION)*, Cambridge, UK, Jul. 2018, pp. 1–8.

[16] D. Krout, M. El-sharkawi, W. J. Fox, and M. Hazen, "Intelligent ping sequencing for multistatic sonar systems," in *2006 9th International Conference on Information Fusion*, Florence, Italy, Jul. 2006, pp. 1–6.

[17] D. W. Krout, G. M. Anderson, E. Hanusa, and B. Jones, "Threat modeling for sensor optimization," in *2013 OCEANS - San Diego*, San Diego, CA, USA, Sep. 2013, pp. 1–4.

[18] P. Tichavsky, C. H. Muravchik, and A. Nehorai, "Posterior Cramer-Rao bounds for discrete-time nonlinear filtering," *IEEE Transactions on Signal Processing*, vol. 46, no. 5, pp. 1386–1396, May 1998.

[19] M. L. Hernandez, T. Kirubarajan, and Y. Bar-Shalom, "Multisensor resource deployment using posterior Cramer-Rao bounds," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 40, no. 2, pp. 399–416, April 2004.

[20] Y. Bar-Shalom, X. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation: Theory, Algorithms and Software*. Wiley, 2002.

[21] B. Ristic, S. Arulampalam, and N. Gordon, "Beyond the Kalman filter: Particle filters for tracking applications," *Artech House*, 2004.

[22] D. W. Krout and T. Powers, "Sensor management for multistatics," in *17th International Conference on Information Fusion (FUSION)*, Salamanca, Spain, July 2014, pp. 1–6.

[23] S. Williams, X. Wang, D. Angley, C. Gilliam, B. Moran, T. Jackson, R. Ellem, and A. Bessell, "Smoothing optimal RRT* trajectories for recovery of an AUV by a moving surface vessel," in *Maritime Situational Awareness Workshop organised by NATO Science and Technology Organization Centre for Maritime Research and Experimentation*, Lerici (La Spezia), Italy, 2018, pp. 1–8.

[24] D. Schuhmacher, B. T. Vo, and B. N. Vo, "A consistent metric for performance evaluation of multi-object filters," *IEEE Transactions on Signal Processing*, vol. 56, no. 8, pp. 3447–3457, August 2008.

[25] J. Barr, O. Harrald, S. Hiscocks, N. Perree, H. Pritchett, and J. W. *et. al.*, "Stone Soup open source framework for tracking and state estimation: enhancements and applications," in *Signal Processing, Sensor/Information Fusion, and Target Recognition XXXI*, L. Grewe, E. Blasch, and I. Kadar, Eds. SPIE, 2022, p. 4.

[26] S. Hiscocks, J. Barr, N. Perree, J. Wright, H. Pritchett, O. Rosoman, M. Harris, R. Gorman, S. Pike, P. Carniglia, L. Vladimirov, and B. Oakes, "Stone Soup: No longer just an appetiser," in *2023 26th International Conference on Information Fusion (FUSION)*, 2023, pp. 1–8.

[27] S. Hiscocks, O. Harrald, J. Barr, N. Perree, L. Vladimirov, and M. H. *et. al.*, "Stone Soup," 1.5V, 2024, Software available at https://zenodo.org/records/14224532.