A Constrained POMDP Formulation and Algorithmic Solution for Radar Resource Management in Multi-Target Tracking

Schöpe, M.I.; Driessen, J.N.; Yarovoy, Alexander

Publication date
2021

Document Version
Final published version

Published in
ISIF Journal of Advances in Information Fusion

Citation (APA)
Schöpe, M. I., Driessen, J. N., & Yarovoy, A. (2021). A Constrained POMDP Formulation and Algorithmic Solution for Radar Resource Management in Multi-Target Tracking. *ISIF Journal of Advances in Information Fusion*, *16*(1), 31.

A Constrained POMDP Formulation and Algorithmic Solution for Radar Resource Management in Multi-Target Tracking

MAX IAN SCHÖPE
HANS DRIESSEN
ALEXANDER G. YAROVOY

The radar resource management problem in a multitarget tracking scenario is considered. The problem is solved using a dynamic budget balancing algorithm. It models the different sensor tasks as partially observable Markov decision processes and solves them by applying a combination of Lagrangian relaxation and policy rollout. The algorithm has a generic architecture and can be applied to different radar or sensor systems and cost functions. This is shown through simulations of two-dimensional tracking scenarios. Moreover, it is demonstrated how the algorithm allocates the sensor time budgets dynamically to a changing environment in a nonmyopic fashion. Its performance is compared with different resource allocation techniques and its computational load is investigated with respect to several input parameters.

Manuscript received October 16, 2020; revised February 4, 2021; released for publication June 30, 2021.

Associate Editor: Jason Williams.
The authors are with the Microwave Sensing, Signals and Systems (MS3), Delft University of Technology, 2628 CD Delft, The Netherlands (E-mail: {m.i.schope; j.n.driessen; a.yarovoy}@tudelft.nl).
This research was partially funded by the Nederlandse Organisatie voor Wetenschappelijk Onderzoek (NWO) through the Integrated Cooperative Automated Vehicles (i-CAVE) project.

I. INTRODUCTION

Recent advances in multifunction radar (MFR) systems led to an increase of their degrees of freedom. As a result, modern MFR systems are capable of adjusting many parameters during runtime. An automatic adaptation of the radar system to changing situations, like weather conditions, interference, or target maneuvers, is often mentioned in the context of MFR and is usually called radar resource management (RRM). It is frequently considered within the broader context of so-called cognitive radar (see, e.g. [7], [10], [15], [19], [26]). Possible applications of these management approaches include automotive scenarios such as autonomous driving or traffic monitoring, (maritime) surveillance, and air traffic control. This paper aims at developing a generic framework and approximately optimal algorithmic solutions for solving RRM problems. Although the focus is on MFR, the approach is not limited to such sensor systems and has a wider applicability.

A. Radar Resource Management

Much of the research on RRM (see e.g. the overview by Hero and Cochran in [21] or by Moo and Ding in [34]) focuses on a single task, e.g. keeping a constant track quality even under target maneuvers. This usually means managing the time budget spent on a certain task. However, MFR systems are usually operating at their sensor time and/or energy budget limit. In such cases, increasing the budget for one task means simultaneously decreasing the budget of the others, inevitably deteriorating their performance. In this paper, part of the RRM problem is therefore seen as a budget or resource balancing act over the individual tasks.

Heuristic solutions have been presented in the past (see, for instance, the overview in [24]), some relying on assigning task priorities and priority-based scheduling. Applying heuristics too early in the design leads to complicated solutions, e.g. nested if-then-else rules. It is not easy to understand what problem is solved within those approaches and whether or not and in what sense the solution is optimal. This usually does not lead to a reusable generic algorithm. In addition, a priority-based scheduler usually does not balance the budget over all tasks but simply schedules the jobs in order of priority (as, e.g. applied in [33] and [39]). When the timeline is fully occupied, it often leaves a set of tasks with the lowest priorities that together do not fit anymore. These approaches do not consider decreasing the time budgets of individual tasks. Furthermore, the determination of the levels of priorities and the rules for assigning them is often not easy and prone to heuristics.

In this paper, the problem is treated as an optimal stochastic control problem. This relies on an explicit formulation of:

- the inference problem that the radar has to solve in terms of dynamic and measurement models,

- the control actions that are available to the sensor, which reflect the degrees of freedom of the MFR mentioned earlier,
- a cost function that reflects the system performance that the user would like to optimize.

To the best of our knowledge, an overall solution to the RRM problem using this approach has not been presented so far. It has been suggested that a truly optimal solution could possibly lead to a significant improvement of the performance of adaptive sensors [20], but that still needs to be illustrated. However, even if the performance would not improve much over heuristic solutions that are carefully tuned to each application, a reusable generic framework will reduce the design effort of RRM solutions. As a consequence, such a framework would reduce the development cost and time and aid in understanding the system behavior.

B. Markov Decision Processes in Resource Management

Markov decision processes (MDPs) and partially observable MDPs (POMDPs) are attractive frameworks for modeling and solving RRM problems. They use a number of states to formulate a dynamic control problem in which the optimal actions can be found through optimizing a cost or reward function. A very good overview of how these schemes can be used for RRM has been published by Charlish et al. in [12].

Those frameworks have been applied to single tasks, for instance, by Charlish and Hoffmann in [13] or by Krishnamurthy in [29]. Both methods optimize the time between consecutive measurement operations. Charlish and Hoffmann are considering a radar tracking example, where the track quality needs to be optimized while Krishnamurthy presents a more general sensor scenario where the measurement performance is optimized regarding false-alarm rate and the quality of the estimate. The former approach applied policy rollout, while the latter used a stochastic dynamic programming algorithm. Two other approaches show how radar actions can be determined by applying reinforcement learning (RL) [38] and deep RL [42] to solve an underlying MDP. In their papers, both Selvi et al. and Thornton et al. are optimizing the sensing strategies for a single target while a communication signal is using the same frequency band. Both publications show that the optimal policy can improve the performance despite the presence of the interferer. We believe that RL is an interesting approach to RRM but that it is often not feasible because of the huge state space that comes with many problem formulations. In such a case, the training of the algorithm would need an enormous amount of data and a lot of computation time.

Constrained (PO)MDPs have been proposed to solve multitask control problems, where the constraint(s) among others can represent the limit on the available resources or budgets for all the tasks. Possible applications are radar networks or single radars with multiple tasks. The computational complexity of these problems is potentially large. It has been suggested to decouple the main optimization problem into smaller and easier-to-solve subproblems by the use of Lagrangian relaxation (LR). One LR approach for sensor networks with an energy constraint on the inter-sensor communication has been published by Williams et al. in [45]. Some notable LR approaches for multitask radar scenarios are, e.g. [46] by Wintenby and Krishnamurthy and [44] by White and Williams. Wintenby and Krishnamurthy apply a Markov chain consisting of performance states for each tracking task and solve it with a combination of LR and approximate dynamic programming. White and Williams assume a discretized state space and a fully observable MDP, which they solve by the use of dynamic programming. In addition to that, Castañón applies LR in combination with a constrained POMDP for multiobject classification in [11]. The chosen POMDP solution method in that approach is the so-called Witness algorithm. Similar to LR, one could also consider the quality of service resource allocation method (Q-RAM) in combination with POMDPs. Although Q-RAM requires an action-space discretization while LR allows the subproblems to be solved analytically, these methods are conceptually very similar. Some interesting approaches using Q-RAM have been shown by Irci et al. in [22] and Charlish et al. in [16] and [14].

Another interesting approach for applying POMDPs for RRM has been introduced by Krishnamurthy and Djonin in [30], where they divide the RRM algorithm into "sensor micromanagement" and "sensor macromanagement." The former is formulated as a POMDP and determines after which time the resource allocation has to be updated. There is always one task that gets a high resource allocation, while the others receive a lower one. The macromanagement, on the other hand, decides which target will get the highest priority and therefore the highest resource allocation. This process is based on the realized cost of the micromanagement and some heuristic rules. Our research, on the other hand, aims at combining micro- and macromanagement. The resource distribution is defined directly through the cost function and without any heuristic functions. In addition to that, the budgets of the tasks can gradually change over time contrarily to the approach of Krishnamurthy and Djonin, where only two different actions exist.

C. The Cost Function

When applying such an RRM approach, the final performance of the sensor system will be determined by the cost function. This is preferred over a heuristic approach; however, it introduces the explicit formulation of such a cost function in the application of the framework. Sometimes it has been suggested that generic measures of performance, such as the information gain, or the Renyi divergence applied to the posterior density of the full state

could be applied (see, e.g. [28], [43]). It is our strong conviction that one single cost function will not meet the desires of different users in different applications with different sensors, targets, and environments.

The development of specific cost functions is important and will be a development task in itself that will require close cooperation with potential users. However, since the primary focus in this paper is on the development of a generic framework for RRM, the development of such user-specific cost functions is out of the scope.

D. Our Approach

In this paper, the RRM problem is considered as a multitask time budget constrained control problem, where the individual tasks are different tracking tasks. Our chosen problem formulation directly leads to the assumption of a constrained POMDP.

Previously, we have already shown the optimal balancing of sensor budgets in a linear time-invariant (LTI) setting by using the optimal steady-state budget balancing (OSB) algorithm [36]. It applies LR to distribute the resources over the different tasks. We have subsequently considered generic dynamical problems by utilizing the POMDP framework and introduced the approximately optimal dynamic budget balancing (AODB) algorithm [37] with a cost function based on the predicted error-covariance of the Kalman filter (KF). We have shown that the results of the AODB algorithm are approximately optimal with respect to the steady-state error-covariance of a KF. The RRM problem was solved nonmyopically by using an online Monte Carlo technique called policy rollout, which stochastically predicts the future.

E. Novelty

In this paper, we compare the performance of the AODB algorithm to several other resource allocation techniques. Furthermore, we investigate its computational load. Compared to our previous papers, we apply the AODB algorithm to a more complete dynamic radar tracking scenario to emphasize its practical value in variable problem settings. In order to do so, we show how the AODB algorithm can be applied to dynamic radar scenarios assuming different measurement types and system parameters.

F. Structure of the Paper

The remainder of this paper is structured as follows. Section II defines the problem as a constrained optimization problem in a POMDP framework, while Section III explains the general application of LR and policy rollout to that problem. Section IV introduces the assumed radar scenario. In Section V, we compare the results of the OSB and the AODB algorithm in a simplified LTI scenario, similar to our work in [37]. In Section VI, we assume a dynamic radar-related scenario, with more realistic parameters than in our previous work. It is solved by applying the AODB algorithm and optimizing both dwell time and revisit interval. Subsequently, an analysis of the algorithm's performance and its computational load is conducted in Sections VII and VIII, respectively. Finally, Section IX contains the conclusions.

II. RRM PROBLEM DEFINITION

A. Motion Model

At every moment in time t, each target considered within this model can be characterized by a state based on its position and velocity in the x and y directions within a two-dimensional Cartesian coordinate system. For target n, this state is defined as

$$\boldsymbol{s}_t^n = [x_t^n \quad y_t^n \quad \dot{x}_t^n \quad \dot{y}_t^n]^T, \tag{1}$$

where x_t^n, y_t^n and $\dot{x}_t^n$, $\dot{y}_t^n$ are the position and velocity of target n in x and y, respectively. The future target state at time $t + \Delta t$ can be calculated following a function

$$\boldsymbol{s}_{t+\Delta t}^n = f_{\Delta t}\left(\boldsymbol{s}_t^n, \boldsymbol{w}_t^n\right), \tag{2}$$

where $s_{t+\Delta t}^n$ is the next following state at time $t + \Delta t$ and $\boldsymbol{w}_t^n \in \mathbb{R}^4$ is the maneuverability noise for target n at time t. The state evolution equation (2) directly defines the evolution probability density function, which is given as

$$p\left(\boldsymbol{s}_{t+\Delta t}^n | \boldsymbol{s}_t^n\right). \tag{3}$$

B. Measurement Model

We assume a sensor that is taking noisy observations of the state $\boldsymbol{s}_t^n$ with sensor action $\boldsymbol{a}_t^n \in \mathbb{R}^m$, where m is the amount of adjustable action parameters. A measurement $\boldsymbol{z}_t^n$ of target n at time t can be characterized by using the measurement function $\mathfrak{h}$ as

$$\boldsymbol{z}_t^n = \mathfrak{h}\left(\boldsymbol{s}_t^n, \boldsymbol{v}_t^n, \boldsymbol{a}_t^n\right), \tag{4}$$

where $\boldsymbol{v}_t^n \in \mathbb{R}^q$ is the measurement noise for target n and q is the amount of measurement parameters. The measurement equation (4) directly defines the measurement probability density function, which can be written as

$$p\left(\boldsymbol{z}_t^n | \boldsymbol{s}_t^n, \boldsymbol{a}_t^n\right). \tag{5}$$

C. Tracking Algorithm

For the tracking scenarios considered in this paper, a tracking algorithm should be chosen that aims at computing the posterior density. For linear systems, a KF can be adopted as an exact solution. For nonlinear systems, possible algorithms are an extended KF (EKF) or a particle filter, for example.

D. Budget Optimization Problem

As mentioned in Section I, the radar sensor is assumed to have a limited maximum budget $\Theta_{\max}$ of any kind. For action $\boldsymbol{a}_t^n$ that is executed for each task n, a certain amount of budget (e.g. time or energy allocations) is required. In an overload situation, the current tasks require more total budget than is available. Thus, the available budget has to be distributed over the tasks in a way that minimizes a cost (e.g. related to the uncertainty of the current situation).

At time t, the optimization problem for N different tasks can be written as

$$\begin{aligned} \underset{\boldsymbol{a}_t}{\text{minimize}} \quad & \sum_{n=1}^{N} c(\boldsymbol{a}_t^n, \boldsymbol{s}_t^n) \\ \text{subject to} \quad & \sum_{n=1}^{N} \Theta_t^n(\boldsymbol{a}_t^n) \leq \Theta_{\max}, \end{aligned} \tag{6}$$

where $\Theta_t^n \in [0, 1]$ is the budget for task n at time t, $c(\cdot)$ is the used cost function, and $\Theta_{\max} \in [0, 1]$ is the maximum available budget (0: no budget assigned, 1: all budget assigned). The definition of an operationally relevant cost function is important to efficiently benefit from these techniques, but is not the focus of this paper. An example of an operationally relevant cost function has been discussed by Katsilieris et al. [25].

III. PROPOSED SOLUTION FOR RRM PROBLEM

A. Distribution of Sensor Budgets Using LR

This paper is partly based on our previous research [36], where we used LR to include the constraints into the cost function. By doing so, the original optimization problem is decoupled into smaller ones, one for each task. This leads to the Lagrangian dual (LD), which needs to be optimized. The LD problem (LDP) is formulated as

$$Z_D = \max_{\lambda_t} \left(\min_{\boldsymbol{a}_t} \left(\sum_{n=1}^{N} \left(c(\boldsymbol{a}_t^n, \boldsymbol{s}_t^n) + \lambda_t \cdot \Theta_t^n \right) \right) - \lambda_t \cdot \Theta_{\max} \right), \tag{7}$$

where $\lambda_t \in \mathbb{R}$ is the Lagrange multiplier for the budget constraint. Due to the sum in the LDP, the minimization problem can be solved for each target n in parallel before updating the Lagrangian multiplier in an iterative process. The exact procedure is shown in [36] and is summarized in the following, where an internal index l is used for the iterations within the LR process.:

1) $l = 0$: set an initial Lagrange multiplier ($\lambda = \lambda_0$).
2) For each task n, minimize the LD with respect to the actions and save resulting $\boldsymbol{a}_l^n$ and Θ_l^n.
3) Choose a subgradient for the Lagrange multiplier as $\mu_l^\lambda = \sum_{n=1}^{N} \Theta_l^n - \Theta_{\max}$.
4) Check if $\mu_l^\lambda \approx 0$ with the desired precision. If it is, stop the process. The current λ_l, $\boldsymbol{a}_l^n$, and Θ_l^n are the final LR solution for λ_t at time t.
5) Set $\lambda_{l+1} = max\{0, \lambda_l + \gamma_l \mu_l^\lambda\}$, where γ_l is the LR step size at time l. This stage is responsible for iteratively maximizing the LD with respect to λ.
6) Go to step 2 and set $l = l + 1$.

Lagrange multipliers and LR have been extensively covered in literature and more information can be found, e.g. in [2], [5], [6], [9], and [31].

B. Definition of a POMDP

A POMDP describes an MDP for which the state cannot be observed directly. Instead, an observation is taken, which generates a probability distribution over the possible states. This is called the belief state. Based on the belief state and the knowledge of the underlying MDP, a POMDP allows us to solve optimization problems nonmyopically, meaning that it takes the expected future into account. In the following, the time is assumed to be discretized in intervals k with length T, which is the time between two consecutive observations.

Generally, a POMDP is defined by the following parameters (see, e.g. [35], [17]):

State space $\boldsymbol{S}$: It consists of all possible states that can be reached within the process, see (1). At time step k, the state is defined as $\boldsymbol{s}_k$. Based on the underlying states and the received observations, the belief state defines a probability distribution over all possible states. It is defined as $\boldsymbol{b}_k$.

Action space $\boldsymbol{A}$: It consists of all possible actions within the process. Each action has a certain cost defined by the cost function. The action at time step k is denoted by $\boldsymbol{a}_k$.

Observation space $\boldsymbol{Z}$: It consists of all possible observations that can be received within the process. An observation at time step k is defined as $\boldsymbol{z}_k$.

Transition probability $\boldsymbol{\Psi}(\boldsymbol{s}_k, \boldsymbol{s}_{k+1}, \boldsymbol{a}_k)$: It is the probability function $p(\boldsymbol{s}_{k+1}|\boldsymbol{s}_k, \boldsymbol{a}_k)$ that defines the probability of transitioning from state $\boldsymbol{s}_k$ to state $\boldsymbol{s}_{k+1}$ given action $\boldsymbol{a}_k$. Note: In this paper, the transition probability does not depend on the action.

Probability of observation $\boldsymbol{O}(\boldsymbol{z}_k, \boldsymbol{s}_{k+1}, \boldsymbol{a}_k)$: It is the probability function $p(\boldsymbol{z}_k|\boldsymbol{s}_{k+1}, \boldsymbol{a}_k)$ that defines the probability of receiving a certain observation $\boldsymbol{z}_k$ when executing action $\boldsymbol{a}_k$ with the resulting state being $\boldsymbol{s}_{k+1}$.

Cost function $c(\boldsymbol{s}_k, \boldsymbol{a}_k)$: It is the immediate cost of executing action $\boldsymbol{a}_k$ in state $\boldsymbol{s}_k$. Note: In this paper, the cost function does not directly depend on the state.

Discount factor γ: It is a discount factor that discounts future time steps with respect to the present. Note: In this paper, the discount factor is always set to $\gamma = 1$.

POMDPs can be solved for finite or infinite horizons. In order to reduce the necessary computational power, a

limited horizon $\mathscr{H}$ is assumed in this paper. The value of $\mathscr{H}$ represents the number of considered measurement time steps into the future. Every time a new budget allocation is calculated, the horizon $\mathscr{H}$ will be reapplied from the current moment in time. This approach is therefore also called a receding horizon.

In [13], Charlish and Hoffmann have written a very clear summary of the general solution of a POMDP, which is used as a base for the following equations. We would like to find the actions that minimize the total cost (value $V_{\mathscr{H}}$ over horizon $\mathscr{H}$). Starting at time step k_0, this can be expressed as

$$V_{\mathscr{H}} = E\left[\sum_{k=k_0}^{k_0+\mathscr{H}} c(\boldsymbol{s}_k, \boldsymbol{a}_k)\right]. \tag{8}$$

Using $C_B(\boldsymbol{b}_k, \boldsymbol{a}_k) = \sum_{\boldsymbol{s}\in\boldsymbol{S}} \boldsymbol{b}_k(\boldsymbol{s}) c(\boldsymbol{s}, \boldsymbol{a}_k)$ being the expected cost given belief state $\boldsymbol{b}_k$, $V_{\mathscr{H}}$ can be written as a so-called value function of the belief state $\boldsymbol{b}_{k_0}$ at time step k_0:

$$V_{\mathscr{H}}(\boldsymbol{b}_{k_0}) = E\left[\sum_{k=k_0}^{k_0+\mathscr{H}} C_B(\boldsymbol{b}_k, \boldsymbol{a}_k)|\boldsymbol{b}_{k_0}\right]. \tag{9}$$

For belief state $\boldsymbol{b}_0$ and taking action $\boldsymbol{a}_0$, the optimal value function is defined according to Bellman's equation [1] as

$$V^*_{\mathscr{H}}(\boldsymbol{b}_0) = \min_{\boldsymbol{a}_0\in\boldsymbol{A}} \left(C_B(\boldsymbol{b}_0, \boldsymbol{a}_0) + \gamma \cdot E\left[V^*_{\mathscr{H}-1}(\boldsymbol{b}_1)|\boldsymbol{b}_0, \boldsymbol{a}_0\right]\right). \tag{10}$$

For very long or infinite horizons, the discount factor can be set to $\gamma < 1$. Using this equation, the optimal policy can be expressed as

$$\pi^*_0(\boldsymbol{b}_0) = \arg\min_{\boldsymbol{a}_0\in\boldsymbol{A}} \left(C_B(\boldsymbol{b}_0, \boldsymbol{a}_0) + \gamma \cdot E\left[V^*_{\mathscr{H}-1}(\boldsymbol{b}_1)|\boldsymbol{b}_0, \boldsymbol{a}_0\right]\right). \tag{11}$$

For each $\boldsymbol{b}_k$ and $\boldsymbol{a}_k$, the optimal so-called Q-value is then defined as

$$Q_{\mathscr{H}-k}(\boldsymbol{b}_k, \boldsymbol{a}_k) = C_B(\boldsymbol{b}_k, \boldsymbol{a}_k) + \gamma \cdot E\left[V^*_{\mathscr{H}-k-1}(\boldsymbol{b}_{k+1})|\boldsymbol{b}_k, \boldsymbol{a}_k\right]. \tag{12}$$

Another way to find the optimal policy is to find the action $\boldsymbol{a}_k$ that minimizes the optimal Q-value:

$$\pi^*_k(\boldsymbol{b}_k) = \arg\min_{\boldsymbol{a}_k\in\boldsymbol{A}}(Q_{\mathscr{H}-k}(\boldsymbol{b}_k, \boldsymbol{a}_k)). \tag{13}$$

It is therefore necessary to calculate the Q-value for all possible actions, which is generally infeasible.

C. Solution Methods for POMDPs

For solving a POMDP, there are both online and offline approaches. The choice of the type of these methods usually depends on the size of the state space. The so-called state-space explosion limits the usefulness of exact offline techniques.

Most offline methods are based on the so-called value iteration (VI), which iteratively calculates the cost/reward values of all possible states. There are exact approaches to VI (e.g. One-Pass algorithm [40]), as well as approximate point-based algorithms (e.g. PBVI or Perseus [41]). The former techniques often lead to very complicated optimization problems, while the latter ones require many grid points within the state space (and therefore a lot of memory and computational effort) in order to converge toward the exact solution. The advantage of offline solutions is that the POMDP is solved only once, and the solution is always valid afterward. Unfortunately, those methods are already infeasible for a small-dimensional state space.

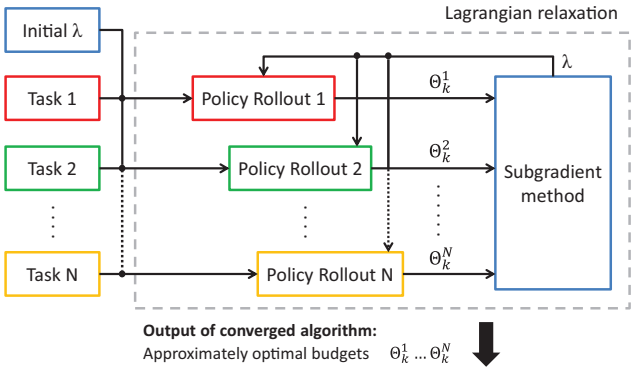

Fig. 1. High-level block scheme of the proposed algorithm.

In contrast to that, online algorithms only solve a small part of the POMDP that is relevant at the current moment. This makes them less accurate, but much easier and faster to compute. Some of the online approaches involve approximate tree methods (see, e.g. the overview in [35]) or Monte Carlo sampling (e.g. policy rollout).

Since an exact and complete solution of the POMDP is usually infeasible in real scenarios, this paper focuses on the implementation of policy rollout as an approximate solution. The general structure of our proposed algorithm is illustrated in Fig. 1. The outputs of the algorithm are the converged budgets for each task.

D. Policy Rollout for POMDPs

The policy rollout technique takes Monte Carlo samples of the expected future, which means that it stochastically explores the possible future actions and the according costs. Within a rollout, observations and belief states are generated from a given initial belief state and a given candidate action. There is a rollout evaluation per action $\boldsymbol{a}$ in the action space $\boldsymbol{A}$. The candidate action is taken in the first step of the rollout, while a so-called base policy π_{base} is used for every following step, until the horizon $\mathscr{H}$ is reached. In each rollout, the total cost is summed up. This procedure is repeated M times and then the summed cost of all M rollouts is averaged. This is the expected cost of the evaluated action. The action that produced the lowest expected cost is chosen for the next time step. It has been shown that policy rollout leads to a policy that is at least as good as the base policy with a very high probability, if enough samples are provided

TABLE I
System Parameters of the Assumed Radar Systems with Respect to the Reference Measurement

System	Measurement	$\sigma^2_{r,0}$ (m^2)	$\sigma^2_{\theta,0}$ (rad^2)	$\sigma^2_{d,0}$ ((m/s)^2)
A	r/θ	625	4e−4	–
B	$r/\theta/d$	2500	2e−4	25

TABLE II
Parameters of Reference Measurement

SNR (SNR$_0$)	RCS (ς_0)	Dwell time (τ_0)	Range (r_0)
1	10 m^2	1 s	50 km

[3]. The choice of the base policy and the amount of samples to be taken are therefore crucial to the performance of the algorithm. The number of samples is equivalent to the number of rollouts M per action that are used to average the cost, or, in other words, one sample is the evaluation of one possible future. Finding a good base policy for a radar scenario is no trivial task. As an example, one could think about using information from previously experienced situations that were similar to the current one. If the executed actions from the last run have been saved, they can be reused again to improve the policy further. This could be considered in the context of RL, for instance. Unfortunately, it is not very likely to experience the exact same situation multiple times if a very big state space is assumed, so the usefulness in such a case is questionable (see also our remark about RL in Section I-B). Another very simple choice of the base policy could be an equal resource allocation to all the tasks. Policy rollout has been covered extensively, e.g. by Bertsekas in [2]–[4].

The policy rollout can be expressed mathematically as shown in (14) and (15). The Q-value is defined as

$$Q^{\pi_{\text{base}}}(\boldsymbol{b}_k, \boldsymbol{a}_k) = C_B(\boldsymbol{b}_k, \boldsymbol{a}_k) + E\left[V^{\pi_{\text{base}}}(\boldsymbol{b}_{k+1})|\boldsymbol{b}_k, \boldsymbol{a}_k\right], \tag{14}$$

where $E[\cdot]$ is the expectation. The best policy can then be found by applying

$$\pi_k(\boldsymbol{b}_k) = \underset{\boldsymbol{a}_k \in \boldsymbol{A}}{\arg\min}(Q^{\pi_{\text{base}}}(\boldsymbol{b}_k, \boldsymbol{a}_k)). \tag{15}$$

Policy rollout does not necessarily lead to the optimal policy. It rather aims at improving the chosen base policy π_{base}.

IV. ASSUMED RADAR SCENARIO

For the rest of this paper, we assume a two-dimensional radar tracking example that will be solved using the AODB algorithm. Measurements are taken in range, angle, and possibly radial velocity. The algorithm is jointly optimizing the revisit interval T (the time between two consecutive measurements) and the dwell time τ (the time that the sensor spends focusing on a target). The algorithm calculates the budgets of all tasks and makes sure that they fit into the time frame, but does not create an explicit schedule. Therefore, the assumed measurements are taken independently of each other and can be overlapping in time. In order to put all tasks into a single timeline, an explicit scheduler needs to be implemented at a lower level. At which moments this budget calculation is performed depends on the preferences of the user. In the following, the assumptions of the assumed radar scenario are explained in more details.

A. Assumed Radar Systems

In the simulations, two different sets of system parameters are assumed as given in Table I. The table shows the measurement noise variances for range ($\sigma^2_{r,0}$), azimuth ($\sigma^2_{\theta,0}$) angle, and radial velocity ($\sigma^2_{d,0}$) with respect to the measurement of a reference target. The parameters of the reference measurement are shown in Table II and are valid for all simulations that are presented in this paper. Radar system A measures range and azimuth only, while system B is able to measure radial velocity as well. The values of the variances in Table I are chosen rather arbitrarily. We do not intend to compare the different radar systems, but rather use them to show how the AODB algorithm can universally be applied to different systems.

B. Velocity Model

The velocity model is assumed to be constant. Between two resource allocation updates, the actions are assumed to stay unchanged. The action vector $\boldsymbol{a}_n \in \mathbb{R}^2$ consists of the dwell time and the revisit interval. The latter defines the time between the measurements of target n. In contrast to our previous publications, in this paper, the revisit interval T_n and the dwell time τ_n are optimized jointly. The revisit intervals with length T_n depend on the targets and are therefore denoted by k_n. Considering this, (2) can explicitly be written as

$$\boldsymbol{s}^n_{k_n+1} = \boldsymbol{F}_n \boldsymbol{s}^n_{k_n} + \boldsymbol{w}^n_{k_n}, \tag{16}$$

with $\boldsymbol{F}_n \in \mathbb{R}^{4\times4}$ defined as

$$\boldsymbol{F}_n = \begin{bmatrix} 1 & 0 & T_n & 0 \\ 0 & 1 & 0 & T_n \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{17}$$

and the maneuverability noise $\boldsymbol{w}^n$ with covariance

$$\boldsymbol{Q}_n = \begin{bmatrix} T_n^4/4 & 0 & T_n^3/2 & 0 \\ 0 & T_n^4/4 & 0 & T_n^3/2 \\ T_n^3/2 & 0 & T_n^2 & 0 \\ 0 & T_n^3/2 & 0 & T_n^2 \end{bmatrix} \sigma^2_{w,n}, \tag{18}$$

where $\sigma^2_{w,n}$ is the maneuverability noise variance of target n.

Because of the nonlinear relationship between measurements and states, an EKF is applied. The corresponding observation matrix $\boldsymbol{H}_{k_n}^n$ is defined as the Jacobian of the measurement transformation function $\boldsymbol{h}$:

$$\boldsymbol{H}_{k_n}^n = \left.\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{s}}\right|_{\boldsymbol{s}_{k_n}^n}. \tag{19}$$

It has dimensions $\boldsymbol{H}_{k_n}^n \in \mathbb{R}^{2\times 4}$ for system A and $\boldsymbol{H}_{k_n}^n \in \mathbb{R}^{3\times 4}$ for system B.

C. Signal-To-Noise Ratio (SNR) Model

In the following examples, we assume sensor measurements in range (r), azimuth (θ), and radial velocity (d). Since the transformation between polar and Cartesian coordinates is nonlinear, the measurement equation in (4) for target n at time step k_n can be defined as

$$\boldsymbol{z}_{k_n}^n = \boldsymbol{h}(\boldsymbol{s}_{k_n}^n) + \boldsymbol{v}_{k_n}^n, \tag{20}$$

where $\boldsymbol{h}(\boldsymbol{s}_{k_n}^n) \in \mathbb{R}^3$ is the measurement transformation function at $\boldsymbol{s}_{k_n}^n$, which for system B is defined as

$$\boldsymbol{h}(\boldsymbol{s}_{k_n}^n) = \left[\sqrt{(x_{k_n}^n)^2+(y_{k_n}^n)^2}\,,\ \operatorname{atan2}\left(y_{k_n}^n, x_{k_n}^n\right),\ \frac{x_{k_n}^n \dot{x}_{k_n}^n + y_{k_n}^n \dot{y}_{k_n}^n}{\sqrt{(x_{k_n}^n)^2+(y_{k_n}^n)^2}}\right]^T \tag{21}$$

and $\boldsymbol{v}_{k_n}^n \in \mathbb{R}^3$ is the measurement noise for target n. The range, azimuth, and radial velocity components of $\boldsymbol{v}_{k_n}^n$ are independent:

$$\boldsymbol{v}_{k_n}^n = [v_{k_n}^{r,n} \quad v_{k_n}^{\theta,n} \quad v_{k_n}^{d,n}]^T, \tag{22}$$

with variances $\sigma_{r,n}^2$, $\sigma_{\theta,n}^2$, and $\sigma_{d,n}^2$. In this paper, the SNR is calculated by using (23), which is based on equations by Koch in [27]:

$$\mathrm{SNR}_{k_n}(\varsigma_n, \tau_n, r_{k_n}^n) = \mathrm{SNR}_0\left(\frac{\varsigma_n}{\varsigma_0}\right)\left(\frac{\tau_n}{\tau_0}\right)\left(\frac{r_{k_n}^n}{r_0}\right)^{-4} e^{-2\Delta\alpha}, \tag{23}$$

where $\Delta\alpha$ is the relative beam positioning error, ς_n is the constant radar cross section (RCS) of the target n, $r_{k_n}^n$ is the distance of target n at time step k_n, and ς_0, τ_0, and r_0 are the corresponding values for a reference target. In (23), the dwell time is used equivalently to the transmitted energy mentioned by Koch. Similar to the approach in [27], the relative beam positioning error is calculated using

$$\Delta\alpha = \frac{\left(\theta_{k_n} - \hat{\theta}_{k_n}\right)^2}{\Gamma^2}, \tag{24}$$

where θ_{k_n} is the real target angle, $\hat{\theta}_{k_n}$ is the predicted target angle in azimuth at time k_n, and Γ is the one-sided beam width in azimuth.

Using (23), the variance of the range, azimuth, and radial velocity measurement noise for target n can be defined as (see, e.g. [32])

$$\sigma_{\bullet,n}^2 = \frac{\sigma_{\bullet,0}^2}{\mathrm{SNR}_{k_n}(\varsigma_n, \tau_n, r_{k_n}^n)}, \tag{25}$$

where $\bullet \in (r, \theta, d)$ and $\sigma_{\bullet,0}^2$ is the measurement noise variance for a reference target 0, as defined in Table I.

Due to the independent measurements, the measurement covariance when using system B can be defined as

$$\boldsymbol{R}_{k_n}^n = \begin{bmatrix} \sigma_{r,n}^2 & 0 & 0 \\ 0 & \sigma_{\theta,n}^2 & 0 \\ 0 & 0 & \sigma_{d,n}^2 \end{bmatrix}. \tag{26}$$

D. Optimization Problem

It is assumed that there are N tracked targets in the environment. The RRM problem can thus be expressed as

$$\begin{aligned} \underset{\boldsymbol{T},\boldsymbol{\tau}}{\text{minimize}} \quad & \sum_{n=1}^{N} c(T_n, \tau_n, \boldsymbol{s}_{k_n}^n) \\ \text{subject to} \quad & \sum_{n=1}^{N} \frac{\tau_n}{T_n} \le \Theta_{\max}, \end{aligned} \tag{27}$$

where $\Theta_{\max} \in [0, 1]$ is the total available budget. The term budget refers to a ratio of dwell time τ to revisit interval T.

Furthermore, every detection is always correctly assigned to the corresponding target.

E. Cost Function

The assumed cost function is constructed from the predicted error-covariance matrix at time step $k_n + 1$. The current predicted error-covariance matrix $\boldsymbol{P}_{k_n|k_n-1}^n \in \mathbb{R}^{4\times 4}$ at time step k_n can be defined for target n as

$$\boldsymbol{P}_{k_n|k_n-1}^n(T_n, \tau_n) = \boldsymbol{F}_n \boldsymbol{P}_{k_n-1|k_n-1}^n(T_n, \tau_n)\boldsymbol{F}_n^T + \boldsymbol{Q}_n, \tag{28}$$

where $\boldsymbol{F}_n$ is the transition matrix with interval length T_n as defined in (17), $\boldsymbol{P}_{k_n-1|k_n-1} \in \mathbb{R}^{4\times 4}$ is the last filtered error-covariance matrix, and $\boldsymbol{Q}_n$ is the maneuverability covariance with interval length T_n as defined in (18). Based on this, another estimation and prediction cycle is applied to the error-covariance. The result is the error-covariance $\boldsymbol{P}_{k_n+1|k_n} \in \mathbb{R}^{4\times 4}$ for time $k_n + 1$:

$$\boldsymbol{P}_{k_n+1|k_n}(T_n, \tau_n) = \boldsymbol{F}_n \boldsymbol{P}_{k_n|k_n}(T_n, \tau_n)\boldsymbol{F}_n^T + \boldsymbol{Q}_n. \tag{29}$$

The cost function that is used in the following sections is based on this expression.

V. LTI EXAMPLE

In this section, a simplified linear time-invariant scenario is assumed in order to investigate if the AODB algorithm converges to the same results as given by the OSB algorithm, which is the optimal solution in this case.

TABLE III
General Simulation Parameters of LTI Scenario

Parameter	Value	Parameter	Value
Precision of LR (δ_{LR}):	0.001	Maximum budget ($\Theta_{\max}$):	1
Action discretization ($\Delta T, \Delta\tau$):	0.0025 s	Budget update (t_{B}):	5 s
Number of rollouts (M):	10	Beam positioning error ($\Delta\alpha$):	0
Rollout horizon ($\mathscr{H}$):	10 steps	Probability of detection (P_{D}):	1

TABLE IV
Initial Target Parameters for LTI Scenario

Target n	x_0^n (km)	y_0^n (km)	$\dot{x}_0^n$ (m/s)	$\dot{y}_0^n$ (m/s)	σ_w^2 (m/s^2)^2	ς_n (m^2)
1	50	0	0	0	25	10
2	50	0	0	0	25	20
3	50	0	0	0	25	30
4	50	0	0	0	25	40
5	50	0	0	0	25	50

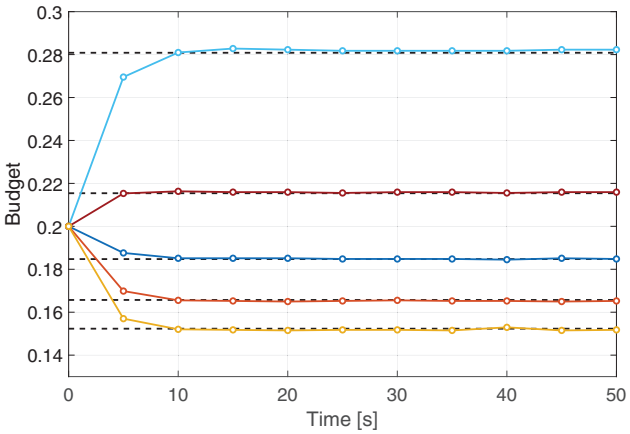

Fig. 2. Budget per task over time after initialization of the AODB algorithm. Solid lines: results from AODB. Dashed lines: optimal steady-state results from OSB. Lines from top to bottom: targets 1–5.

A. General Simulation Parameters

We consider radar system A, as mentioned in Table I. For this simple example, no beam positioning error is taken into account, e.g. due to a very wide beam by using an MFR with a phased array antenna applying digital beamforming (DBF) on receive. The probability of detection is assumed to be 1. The implemented base policy is simply to apply the evaluated action in every step of the policy rollout. Therefore, $\pi_{\mathrm{base}} = \boldsymbol{a}$. A constant LR step size is applied in all simulations. Within the policy rollout, the expected future cost is simulated over the defined horizon for each possible action. The action that produces the lowest expected cost will be chosen for the measurements during the next time steps. No additional random movement (process noise) is considered within the policy rollout. For the simulations in this section, the sum of the predicted error-covariance for the position in the x and y directions is applied as a cost function. Because we want to avoid choosing parameters that are impractical in a real application, an extra term is added that penalizes small values of T. Using (29), this can be expressed as

$$c(T_n, \tau_n) = \mathrm{trace}\left(\boldsymbol{E}\boldsymbol{P}_{k_n+1|k_n}(T_n, \tau_n)\boldsymbol{E}^T\right) + \frac{1000}{(T_n)^2}, \quad (30)$$

where

$$\boldsymbol{E} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \quad (31)$$

is the selection matrix that selects the upper left two-by-two submatrix from the error-covariance matrix.

Table III shows general simulation parameters. The initial Lagrange multiplier value is set to 1. The budgets are recalculated every $t_{\mathrm{B}} = 5$ s. The base policy is executing the evaluated action in every step of the policy rollout horizon ($\pi_{\mathrm{base}} = \boldsymbol{a}$). Within the policy rollout, the expected future is simulated and evaluated for each possible action. The radar is always positioned at the origin of the Cartesian coordinate system.

B. Comparison of OSB and AODB

In order to prove the validity of the proposed AODB algorithm, a comparison is conducted with the OSB algorithm, as proposed in [36]. The OSB algorithm calculates the optimal steady-state error-covariance given a revisit interval T and a dwell time τ by using equations by Kalata in [23] and by Gray and Murray in [18]. It is used as explained in [36] with the general simulation parameters from Table III.

For the comparison, system A and five target tracking tasks are considered with the parameters shown in Table IV. The revisit interval T and the dwell time τ are discretized in steps of 0.0025 s. It is assumed that the budget values are recalculated every 5 s. In between, measurements of the targets are taken with the previously calculated revisit intervals T_n and dwell times τ_n. The tracks are assumed to be initialized at the beginning of the simulation.

Since the steady-state solution of the OSB algorithm is only valid for a single dimension, we assume that the targets are all positioned at the same position and the system knows the exact azimuth angle. All targets are static and only the RCS is considered to be different.

The simulation results are shown Fig. 2. It can be seen that the budget allocations $\Theta^n = \tau_n/T_n$ converge to results that are very close to the values that have been determined with the OSB algorithm.

Theoretically, the AODB algorithm should work with any number of tasks. In order to demonstrate that, the above simulation has been repeated with 10 tasks. Equivalent to targets 1–5, the RCS values of targets 6–10

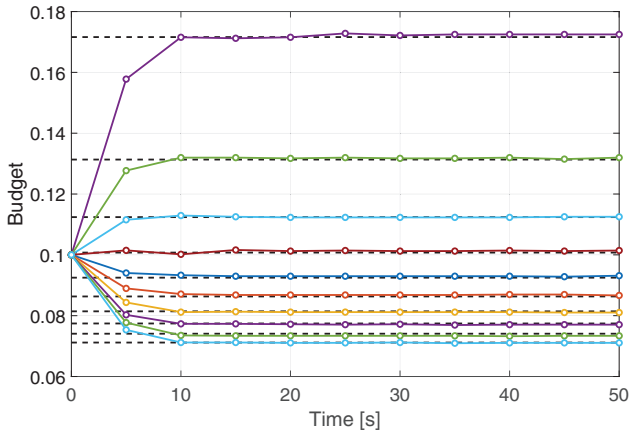

Fig. 3. Budget per task over time after initialization of the AODB algorithm. Same simulation as for Fig. 2, but with ten tracking tasks. Lines from top to bottom: targets 1–10.

TABLE V
General Simulation Parameters of Dynamic Scenario

Parameter	Value	Parameter	Value
Precision of LR (δ_{LR}):	0.001	Maximum budget ($\Theta_{\max}$):	1
Action discretization ($\Delta T, \Delta\tau$):	0.0025 s	Budget update (t_{B}):	5 s
Number of rollouts (M):	5	Beam positioning error ($\Delta\alpha$):	0
Rollout horizon ($\mathscr{H}$):	15 steps	Probability of detection (P_{D}):	1

are increasing in steps of 10 m^2. Fig. 3 shows the approximately optimal budget distribution.

VI. DYNAMIC RADAR EXAMPLE

In this section, the performance of the AODB algorithm is investigated in a more realistic radar-related example with different system parameters.

A. General Simulation Parameters

The cost function as introduced in (30) is applied. Table V shows general simulation parameters for these simulations. The initial Lagrange multiplier value is set to 1. The budgets are recalculated every $t_{\mathrm{B}} = 5$ s and measurements are taken in between with the current calculated resource allocations. The base policy is executing the evaluated action in every step of the policy rollout horizon ($\pi_{\mathrm{base}} = \boldsymbol{a}$). Within the policy rollout, the expected future is simulated and evaluated for each possible action. The radar is always positioned at the origin of the Cartesian coordinate system.

B. Dynamic Radar Scenario for $P_{\mathrm{D}} = 1$

A dynamic scenario with five moving targets is considered in this simulation. The initial target parameters

TABLE VI
Initial Target Parameters for Dynamic Scenario

Target n	x_0^n (km)	y_0^n (km)	$\dot{x}_0^n$ (m/s)	$\dot{y}_0^n$ (m/s)	σ_w^2 $(\mathrm{m/s}^2)^2$	ς_n (m^2)
1	12	10	9	−15	25	20
2	12	15	−30	15	25	20
3	7	11	45	30	64	10
4	19	2	−35	0	64	10
5	10	11	−20	−25	64	10

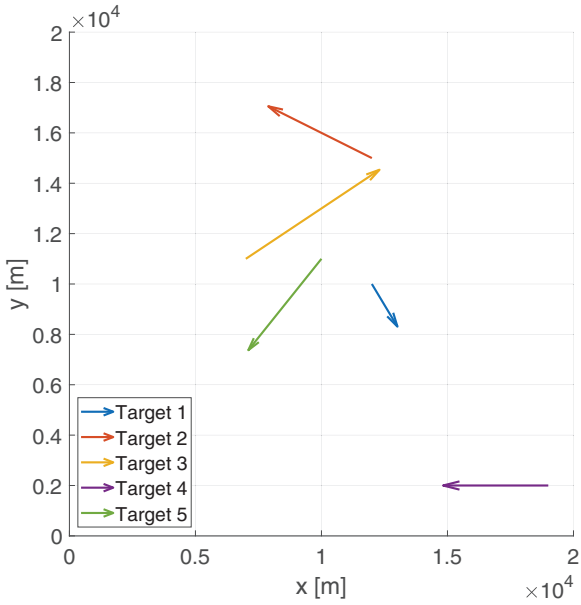

Fig. 4. Trajectories of targets in a dynamic scenario.

are given in Table VI and are valid at the moment when the corresponding track is started. Their trajectories are shown in Fig. 4. The simulation is conducted with systems A and B separately. As in the LTI simulations of Section V, no beam positioning error is taken into account, e.g. due to a very wide beam by using an MFR with a phased array antenna applying DBF on receive. The probability of detection is assumed to be 1. A horizon of $\mathscr{H} = 15$ is assumed. Targets 1–4 are tracked from the beginning, while target 5 joins as a new track after 25 s. After 60 s, the total budget is reduced to $\Theta_{\max} = 0.9$. The reason for this could be that an operator manually assigned 10% of the budget to another task, for instance. At 90 s, the maneuverability variance of target 1 increases by a factor of 36 to a value of 900 $(\mathrm{m/s}^2)^2$, which is known to the system in advance, for instance, through some knowledge of the environment. The simulation results for system A can be found in Figs. 5 and 6, where the former shows the resource distribution over the tasks over time and the latter shows the amount of LR iterations that was needed for convergence. The corresponding simulation results for system B are shown in Figs. 7 and 8.

The algorithm manages to calculate the budget for both systems, while adjusting to unknown and known changes. Before the known variance change at 90 s, the

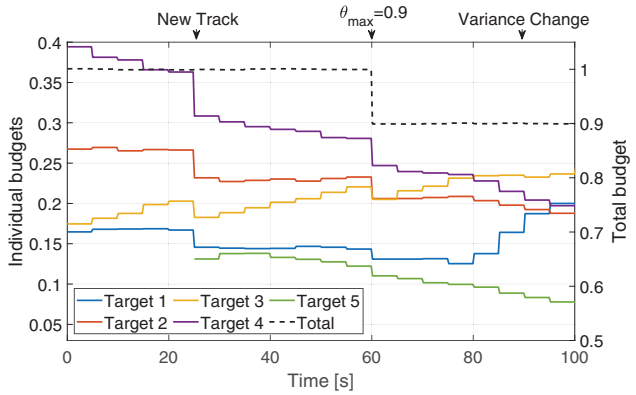

Fig. 5. Dynamic scenario simulation using radar system A.

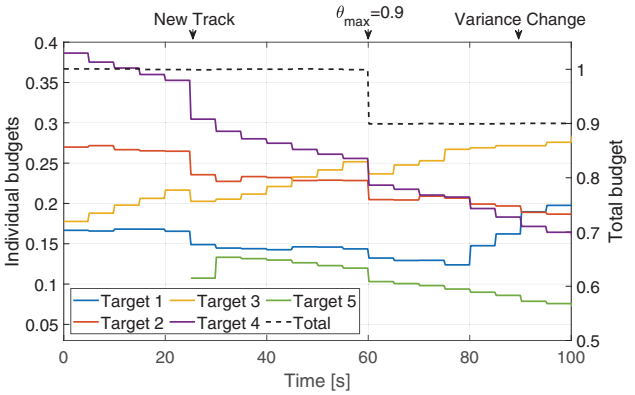

Fig. 7. Dynamic scenario simulation using radar system B.

algorithm already gradually increases the budget for target 1. The algorithm delivers very similar but still different solutions for the two systems. It can be seen that the amount of LR iterations needed until convergence stays low, unless bigger changes in the situation take place. For the chosen parameters, the maximum is 66 iterations for a single resource allocation calculation assuming system B. Using system A leads to similar peak values.

Apart from the impact of the three mentioned sudden changes that are applied to the system, it is also obvious that there seems to be a certain dependence of the budgets on the range. While the budget assigned to targets 1 and 2 stays roughly constant in between different events, target 3 gets an increasing amount of resources assigned, while the resources of targets 4 and 5 are decreasing. The reason for this is that targets 1 and 2 are moving roughly perpendicular to the radar, while target 3 is moving away from it and targets 4 and 5 are moving toward it. In Section VI-D, this effect is investigated with an extra simulation.

C. Dynamic Radar Scenario for $P_D < 1$

In a real situation, a low SNR can lead to missed detections. In addition to that, the used radar system might not have the capability to transmit with a wide beam and apply DBF on receive. Therefore, another simulation is presented in this subsection that takes into account a probability of detection based on the calculated SNR and the beam positioning error. The scenario is identical with the one shown in Section VI-B, and apart from the probability of detection and the beam positioning error, all values in Table V are applied. The SNR is calculated using (23) and taking into account a beam width of 2°. In addition to that, a measurement in the simulation as well as in the policy rollout is only generated with the probability of detection [27]

$$P_{\mathrm{D},k_n} = P_{\mathrm{FA}}^{\frac{1}{1+\mathrm{SNR}_{k_n}}}, \tag{32}$$

where $P_{\mathrm{FA}} = 10^{-4}$ is the constant probability of false alarm. It is assumed that the false alarms have no influence on the tracks. The result of this simulation assuming system A can be found in Figs. 9 and 10.

It can be seen that the resulting budget allocations are less smooth than in the simulations assuming $P_{\mathrm{D}} = 1$. Still, the AODB algorithm leads to comparable results despite the fact that some of the probabilities of detection are quite low.

D. Analysis of the Impact of the Chosen Cost Function

To show the impact of the range on the resource distribution by the AODB algorithm, another simulation has been conducted with three targets. Target 1 has the initial parameters $x_0 = 6$ km, $y_0 = 6$ km, $\dot{x}_0 = 50$ m/s, and $\dot{y}_0 = 50$ m/s. Targets 2 and 3 are static at positions $x = 12.4$ km, $y = 9$ km and $x = 8.4$ km, $y = 9.2$ km, respectively. The simulation result is presented in Fig.

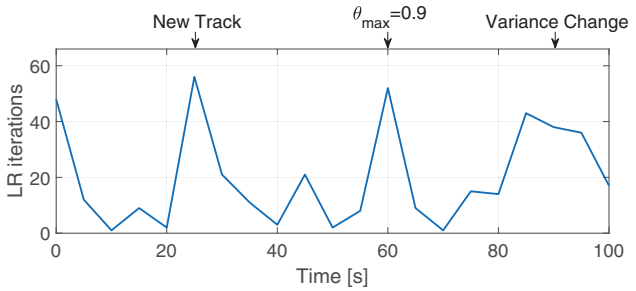

Fig. 6. Number of LR iterations for a dynamic scenario simulation using radar system A.

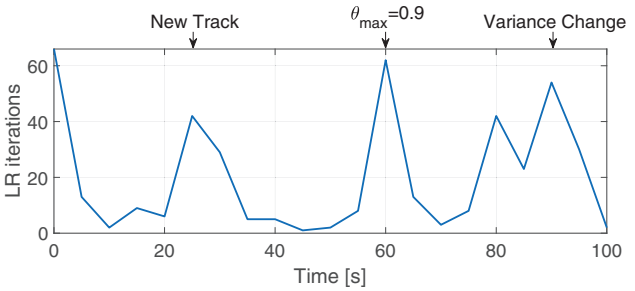

Fig. 8. Number of LR iterations for a dynamic scenario simulation using radar system B.

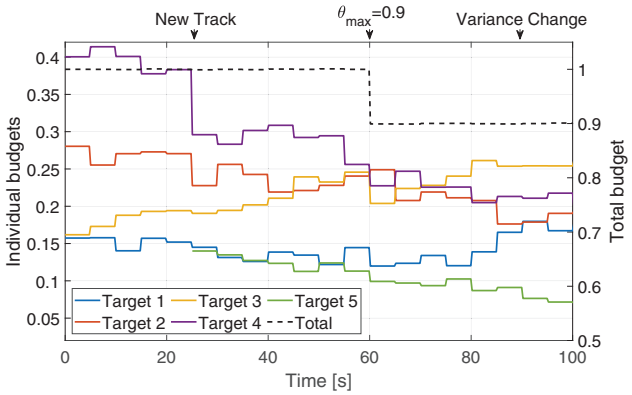

Fig. 9. Dynamic scenario simulation using radar system A with $P_D < 1$.

11 and shows that the budget assigned to target 1 is increasing with growing target distance from the radar, while the budget assigned to the other targets is decreasing. This behavior is expected but does not represent what is typically desired or expected for a radar application.

VII. ANALYSIS OF PERFORMANCE

In the following subsections, we will take a closer look at the general performance of the AODB algorithm with respect to other resource allocation methods.

The assumed scenario is the same as in Section VI, so the radar and target parameters are identical to Tables I, V, and VI. For the following simulations, we consider one implementation of the AODB algorithm and three other strategies using radar system A. The cost evaluation is done for two cases, firstly for $P_D = 1$ and secondly for $P_D < 1$, based on the SNR including a beam positioning error as presented in Section VI-C.

It is generally difficult to judge the performance of RRM algorithms in theory, because it depends on the specific situation and the specific mission where they are applied in. Depending on the user of the radar system, there might be different views on the different parameters. It is possible to show that an approach optimizes the resource distribution according to the chosen cost function, but if the cost function is not well designed, the tracking, detection, or classification performance can

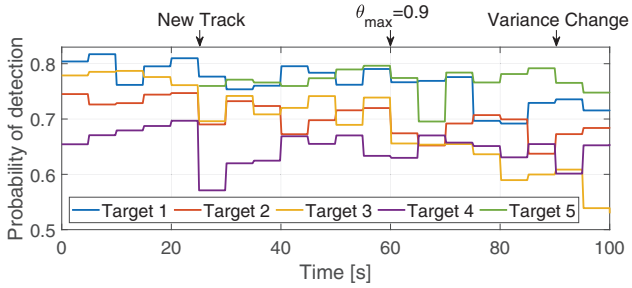

Fig. 10. Average probability of detection per budget update interval for the dynamic scenario with $P_D < 1$.

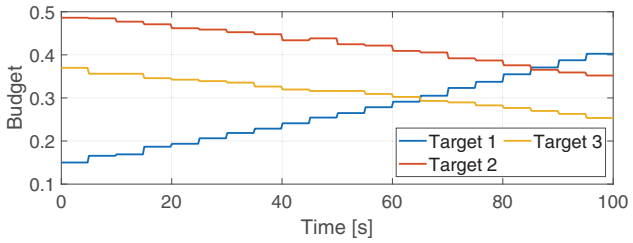

Fig. 11. Budget allocation for two static and one moving target.

still be unsatisfying. Therefore, we only focus on the expected cost in this section.

The techniques that are compared to each other are as follows:

- **Random policy:** For a given revisit interval $T = 1.2$ s, randomly divide the available resources among all tasks.
- **Equal policy:** For a given revisit interval $T = 1.2$ s, the available budget is always distributed equally to all tasks.
- **Unequal policy:** For a given revisit interval $T = 1.2$ s, target 1 gets more resources assigned than the other targets. The remaining resources are distributed equally over targets 2–5.
- **AODB15:** Nonmyopic AODB algorithm, assigning resources using policy rollout ($\mathscr{H} = 15, M = 5$).

Figs. 12 and 13 show how the expected cost develops over time for the different techniques that are mentioned above. For the heuristic methods, the future expected cost during a horizon of $\mathscr{H}$=15 has been evaluated stochastically assuming the chosen action, equivalently to the policy rollout. One can see how the AODB clearly minimizes the cost compared to the other techniques for both $P_D = 1$ and $P_D < 1$.

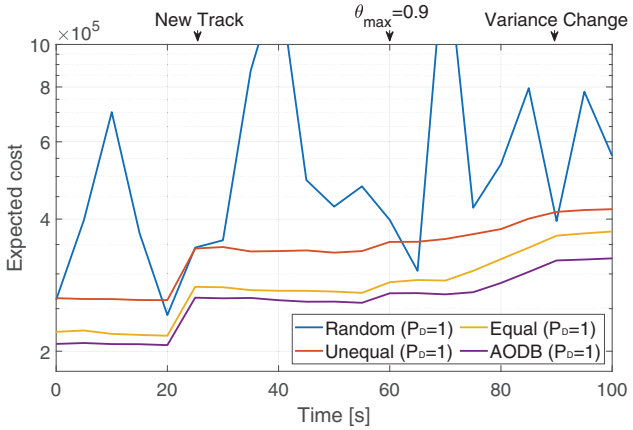

Fig. 12. Comparison of the expected cost for different resource distribution methods assuming radar system A and $P_D = 1$. Note that the cost is plotted in a logarithmic scale.

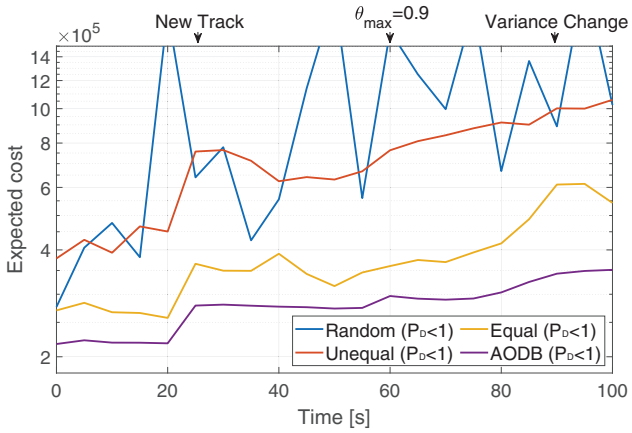

Fig. 13. Comparison of the expected cost for different resource distribution methods assuming radar system A and $P_D < 1$. Note that the cost is plotted in a logarithmic scale.

TABLE VII
General Simulation Parameters for Computational Load Analysis

Parameter	Value	Parameter	Value
Precision of LR (δ_{LR}):	0.01	Maximum budget (Θ_{max}):	1
Action discretization ($\Delta T, \Delta\tau$):	0.0035 s	Number of simulations:	10
Number of rollouts (M):	2	Beam positioning error ($\Delta\alpha$):	0
Rollout horizon ($\mathscr{H}$):	2 steps	Probability of detection (P_D):	1

VIII. ANALYSIS OF COMPUTATIONAL LOAD

In this section, the computational load of the AODB algorithm is investigated. It should be noted that the current version of the algorithm has not been derived with high efficiency in mind. The following results should be seen as indications, since the process can still be optimized.

The computational load of the algorithm has been investigated with respect to the following parameters:

- amount of tracking tasks,
- step size of LR,
- desired precision of results,
- initial value of the Lagrange multiplier,
- rollout length.

In the following, simulation results are shown based on a single budget calculation. This means that we look at the way the LR converges to its final result based on the above parameters. To generate the figures, the results of ten simulations have been averaged. The general simulation parameters are shown in Table VII. Those parameters are valid for all following simulations, except for the currently evaluated parameter. For that one, a sweep over different values is applied, which is specified in the corresponding subsection. We assume a fixed action space that is the same for each calculation in the parameter sweep. The chosen system for these simulations produces measurements in range and angle (system A) and the target parameters are the same as in Section VI (see Table VI). The initial Lagrange multiplier value is set to 1. In addition to that, the cost function as introduced in (30) is used for all following simulations. In the following figures, we show normalized times and normalized LR iterations numbers. This means that each data graph is normalized w.r.t. its maximum value. This is done in order to emphasize that the capability of the hardware and the choice of the general input parameters are not relevant for the discussion of the results.

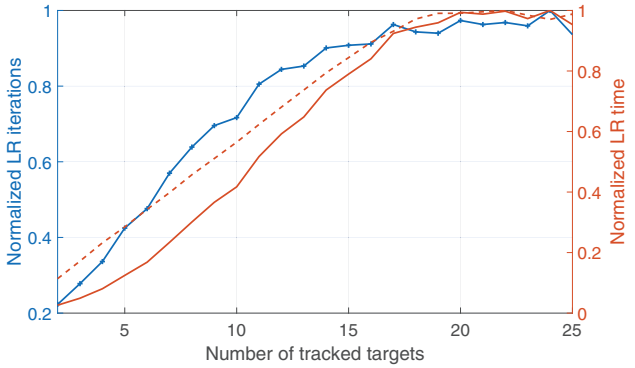

Fig. 14. Convergence for different numbers of tracking tasks. Number of LR iterations (blue with crosses) and total time (solid red) needed for convergence, as well as time per LR iteration (dashed red).

A. Influence of Number of Tasks on AODB

The following simulation shows the influence of an increasing number of tasks on the computational load and execution time of the AODB algorithm. Using the above-mentioned parameters, 24 different simulations have been conducted for 2–25 tracking tasks. The initial Lagrange multiplier value is 1 and the chosen constant LR step size is 8000. Therefore, it is assumed that there is no prior knowledge about the optimal Lagrange multiplier. The results of this simulation can be seen in Fig. 14. It can be seen that the amount of iterations, the total time until the LR converges, and the time needed for each LR iteration are increasing approximately linearly for a rising number of tracked targets, until the increase slows down for larger amounts of targets of 15 and more.

B. Influence of LR Step Size on AODB

In this subsection, a simulation shows the influence of an increasing LR step size on the computational load and execution time of the AODB algorithm. We consider 5 tracking tasks and 50 step sizes between 250 and 20 000, while the initial Lagrange multiplier value is 1. The results of this simulation can be found in Fig. 15. It can be seen that the amount of LR iterations needed and the time until convergence are decreasing exponentially.

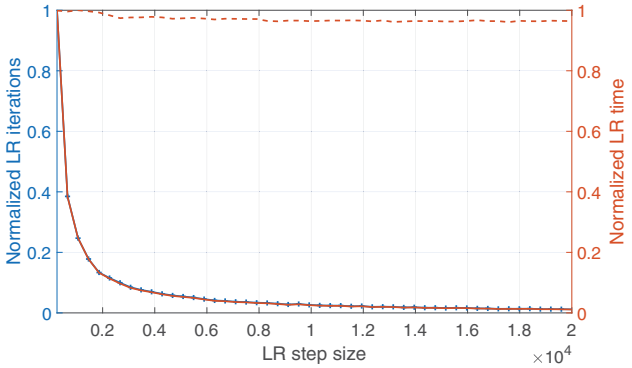

Fig. 15. Convergence for different LR step sizes. Number of LR iterations (blue with crosses) and total time (red) needed for convergence, as well as average time per LR iteration (dashed red).

The average time per LR iteration stays approximately constant.

C. Influence of LR Precision on AODB

The following simulation shows the influence of different LR result precisions on the computational load and execution time of the AODB algorithm. We consider 5 tracking tasks and 50 precision values between 0.001 and 0.2. The results of this simulation can be found in Fig. 16. The initial Lagrange multiplier value is 1 and the chosen constant step size is 8000. It can be seen that the amount of LR iterations and the total LR convergence time are decreasing roughly exponentially. The average time per LR iteration stays approximately constant.

D. Influence of Initial Lagrange Multiplier Value on AODB

This simulation shows the influence of different initial Lagrange multiplier values on the computational load and execution time of the AODB algorithm. We consider 5 tracking tasks and 50 initial Lagrangian multiplier values between 1 and 100 000. The chosen constant step size is 8000. The results of this simulation can be found in Fig. 17. It can be seen that the amount of

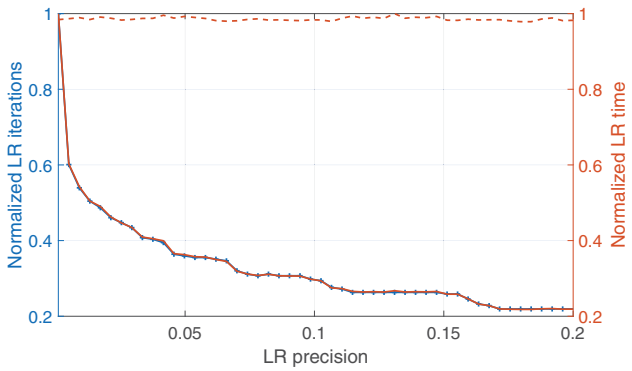

Fig. 16. Convergence for different LR result precisions. Number of LR iterations (blue with crosses) and total time (red) needed for convergence, as well as average time per LR iteration (dashed red).

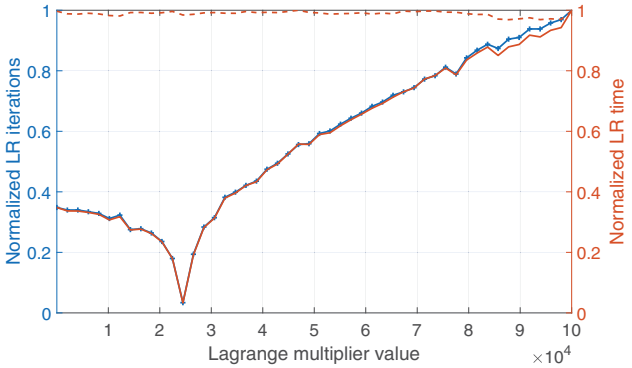

Fig. 17. Convergence for different initial Lagrange multiplier values. Number of LR iterations (blue with crosses) and total time (red) needed for convergence, as well as average time per LR iteration (dashed red).

LR iterations and the LR convergence time have a clear minimum at about 24 000. This is the best starting value, because it allows for the fastest convergence. The average time per LR iteration stays approximately constant.

E. Influence of Rollout Horizon Lengths on AODB

The following simulation shows the influence of different policy rollout horizon lengths on the computational load and execution time of the AODB algorithm. We consider five tracking tasks and the rollout length to vary from 1 to 25. The initial Lagrange multiplier value is 1 and the chosen constant step size is 8000. The results of this simulation can be found in Fig. 18. It can be seen that the amount of LR iterations increases fast in the beginning, before slowly decreasing again for horizon lengths of 6 and longer. The total time needed increases approximately linearly, as well as the time needed per LR iteration.

F. Conclusions on Computational Load

Based on the simulation result of the previous subsections, some conclusions can be made regarding the choice of the input parameters. They will be summarized in the following paragraphs.

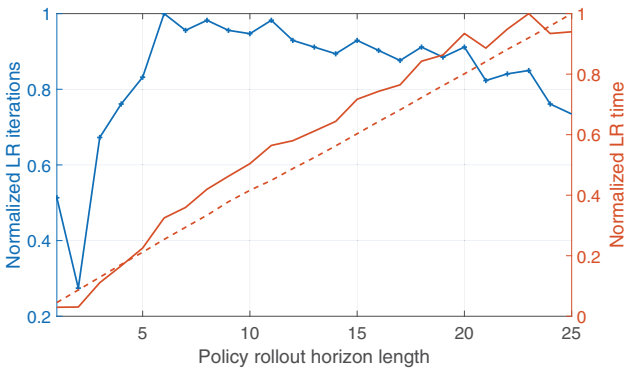

Fig. 18. Convergence for different rollout horizon lengths. Number of LR iterations (blue with crosses) and total time (red) needed for convergence, as well as average time per LR iteration (dashed red).

Number of targets and initial Lagrange multiplier value: Both the necessary number of LR iterations and the total LR convergence time increase with an increasing number of tracking tasks. Unfortunately, it is generally not possible to influence the amount of tasks at will. However, the effect of increasing convergence time can be reduced by choosing the appropriate initial Lagrange multiplier value. We found that there is a distinct minimum in the number of LR iterations before convergence (see Fig. 17). The minimum convergence time, which is equivalent to a single LR iteration, is attained when that Lagrange multiplier value is chosen as the initial value. It is interesting to see that initial Lagrange multiplier values that are bigger than the optimal value lead to longer computations compared to values smaller than the optimum. If some prior knowledge about the Lagrange multiplier value is available (e.g. from the previous budget calculation), this can tremendously decrease the convergence time, if the situation has not changed too much since. Boyd et al. have labeled this approach a "Warm Start" [8].

LR step size and precision of LR result: Increasing LR step size and decreasing precision both lead to a decreasing number of LR iterations and time until convergence, while the time needed for one LR iteration stays more or less constant. Generally, it is useful to choose a rather big LR step size, but if it is chosen too big with respect to the precision and the action-space discretization, the algorithm might not converge but oscillate around the minimum. If the desired results lie in a local minimum instead of the global one, the algorithm might miss that minimum entirely, in case the step size is chosen too big. Therefore, choosing a constant step size is probably not the best solution and adaptive step sizes could increase the performance. There is more freedom to choose the precision, but one should keep in mind that a lower precision will lead to a less accurate result, which can lead to not precisely meeting the maximum budget constraint.

Policy rollout horizon length: Although this paper does not investigate the advantages of choosing different horizon lengths for the policy rollout, it was chosen to examine its impact on the computational load for the sake of completeness. In the future, the impact of the horizon needs to be studied in more detail. Increasing the horizon length leads to an almost linear increase of the time per LR iteration. Very short horizons seem to lead to very low numbers of LR iterations until convergence. For horizon lengths longer than 2, the number of LR iterations increases very quickly, although for horizons longer than 6, it slightly decreases again. The total LR convergence time increases with growing rollout length (see Fig. 18). It is therefore reasonable to choose the shortest horizon necessary. It needs to be kept in mind that this is a trade-off with an impact on the track performance, so a longer horizon can potentially improve the mission performance further.

IX. CONCLUSIONS

In this paper, we have developed a framework and proposed approximately optimal algorithmic solutions for solving RRM problems and shown applicability of the algorithm to a dynamic multitarget tracking scenario. The proposed framework models the different sensor tasks as constrained POMDPs and solves them by applying a combination of Lagrangian relaxation and policy rollout. In contrast to previous work where LTI scenarios were considered, this paper focuses on dynamic situations with different parameters. We believe that the proposed solution is a step toward a truly generic framework.

In a simple radar tracking scenario, the dwell time and the revisit interval were optimized using a cost function based on the predicted position error-covariance that was computed using the EKF.

It was shown that the AODB algorithm budget allocations are close to the optimal steady-state solution in an LTI setting. Furthermore, the simulation results show that the AODB algorithm can be applied to different systems, and it was pointed out how it adjusted itself to known as well as unknown situational changes in a dynamic scenario.

The presented cost function leads to a larger budget being given to tracking tasks with higher uncertainty. At first glance, this may seem to be fully appropriate; however, in radar this means that more budget will be assigned to targets at longer range. Thus, a simple error-covariance-based cost function will not always suit practical radar applications.

An analysis of the performance of the algorithm has also been conducted by comparing the optimized cost to other resource distribution methods. It was found that the AODB always led to the lowest cost values compared to the other considered techniques. Finally, the computational load of the algorithm was investigated. Based on those results, suggestions about a good choice of input parameters have been presented.

In future work, we will investigate the usage of the AODB algorithm in a combined tracking and classification scenario. Furthermore, we will investigate the impact of choosing different horizon lengths and its impact on the cost and the track accuracies. Finally, we will have a closer look at the convergence of the algorithm and how its efficiency can be improved.

REFERENCES

[1] R. E. Bellman
Dynamic Programming. New Jersey, NJ, USA: Princeton Univ. Press, 1957.

[2] D. P. Bertsekas
Constrained Optimization and Lagrange Multiplier Methods. Nashua, NH, USA: Athena Scientific, 1996.

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

[4] D. P. Bertsekas, J. N. Tsitsiklis, and C. Wu Rollout "Algorithms for combinatorial optimization," *J. Heuristics*, vol. 3, no. 3, pp. 245–262, Dec. 1997.
[5] D. Bertsimas and J. N. Tsitsiklis Introduction to Linear Optimization. Nashua, NH, USA: Athena Scientific, 1997.
[6] S. S. Blackman and R. Popoli Design and Analysis of Modern Tracking Systems. London, U.K.: Artech House, 1999.
[7] M. Bockmair, C. Fischer, M. Letsche-Nuesseler, C. Neumann, M. Schikorr, and M. Steck "Cognitive radar principles for defence and security applications," *IEEE Aerosp. Electron. Syst. Mag.*, vol. 34, no. 12, pp. 20–29, Dec. 2019.
[8] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, 2011.
[9] S. Boyd and L. Vandenberghe *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.
[10] S. Brüggenwirth, M. Warnke, S. Wagner, and K. Barth "Cognitive radar for classification," *IEEE Aerosp. Electron. Syst. Mag.*, vol. 34, no. 12, pp. 30–38, Dec. 2019.
[11] D. A. Castañón "Approximate dynamic programming for sensor management," in *Proc. 36th IEEE Conf. Decis. Control*, Dec. 1997, pp. 1202–1207.
[12] A. Charlish, K. Bell, and C. Kreucher "Implementing perception–action cycles using stochastic optimization," in *Proc. IEEE Radar Conf.*, Sep. 2020.
[13] A. Charlish and F. Hoffmann "Anticipation in Cognitive Radar using Stochastic Control," in *Proc. 2015 IEEE Radar Conf.*, May 2015, pp. 1692–1697.
[14] A. Charlish and F. Hoffmann "Cognitive radar management," in *Radar Techniques and Applications, Volume 2—Waveform Diversity and Cognitive Radar, and Target Tracking and Data Fusion*. London, U.K.: Scitech Publishing, 2017, pp. 157–193.
[15] A. Charlish, F. Hoffmann, and I. Schlangen "The development from adaptive to cognitive radar resource management," *IEEE Aerosp. Electron. Syst. Mag.*, vol. 35, no. 6, pp. 8–19, Jun. 2020.
[16] A. Charlish, K. Woodbridge, and H. Griffiths "Phased array radar resource management using continuous double auction," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 51, no. 3, pp. 2212–2224, Jul. 2015.
[17] E. K. P. Chong, C. M. Kreucher, and A. O. Hero "Partially observable markov decision process approximations for adaptive sensing," *Discrete Event Dyn. Syst.*, vol. 19, no. 3, pp. 377–422, Sep. 2009.
[18] J. E. Gray and W. Murray "A derivation of an analytic expression for the tracking index for the alpha-beta-gamma filter," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 29, no. 3, pp. 1064–1065, Jul. 1993.
[19] S. Haykin "Cognitive radar: a way of the future," *IEEE Signal Process. Mag.*, vol. 23, no. 1, pp. 30–40, Jan. 2006.
[20] A. O. Hero, D. A. Castañón, D. Cochran, and K. Kastella *Foundations and Applications of Sensor Management*. New York, NY, USA: Springer Publishing, 2008.
[21] A. O. Hero and D. Cochran "Sensor management: past, present, and future," *IEEE Sensors J.*, vol. 11, no. 12, pp. 3064–3075, Dec. 2011.
[22] A. Irci, A. Saranli, and B. Baykal "Study on Q-RAM and feasible directions based methods for resource management in phased array radar systems," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 46, no. 4, pp. 1848–1864, Oct. 2010.
[23] P. R. Kalata "The tracking index: a generalized parameter for alpha–beta and alpha–beta–gamma target trackers," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 20, no. 2, pp. 174–182, Mar. 1984.
[24] F. Katsilieris "Sensor Management for Surveillance and Tracking: An Operational Perspective," Ph.D. thesis, Delft, The Netherlands, 2015.
[25] F. Katsilieris, H. Driessen, and A. Yarovoy "Threat-based sensor management for target tracking," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 51, no. 4, pp. 2772–2785, Oct. 2015.
[26] R. Klemm, H. Griffiths, and W. Koch *Novel Radar Techniques and Applications, Volume 2 —Waveform Diversity and Cognitive Radar, and Target Tracking and Data Fusion*. London, U.K.: Scitech Publishing, 2017.
[27] W. Koch "On adaptive parameter control for phased-array tracking," in *Proc. SPIE's Int. Symp. Opt. Sci., Eng., Instrum.*, vol. 3809, 1999, pp. 444–455.
[28] C. Kreucher, K. Kastella, and A. O. Hero "Sensor management using an active sensing approach," *Signal Process.*, vol. 85, no. 3, pp. 607–624, Mar. 2005.
[29] V. Krishnamurthy "POMDP Sensor Scheduling with Adaptive Sampling," in *Proc. 17th Int. Conf. Inf. Fusion*, Jul. 2014, pp. 1–7.
[30] V. Krishnamurthy and D. V. Djonin "Optimal threshold policies for multivariate POMDPs in radar resource management," *IEEE Trans. Signal Process.*, vol. 57, no. 10, pp. 3954–3969, Oct. 2009.
[31] F. L. Lewis, D. L. Vrabie, and V. L. Syrmos *Optimal Control*. Hoboken, NJ, USA: Wiley, 2012.
[32] H. Meikle *Modern Radar Systems*. Norwood, MA, USA: Artech House, 2008.
[33] H. S. Mir and A. Guitouni "Variable dwell time task scheduling for multifunction radar," *IEEE Trans. Automat. Sci. Eng.*, vol. 11, no. 2, pp. 463–472, Apr. 2014.
[34] P. W. Moo and Z. Ding *Adaptive Radar Resource Management*. London, U.K.: Academic Press, 2015.
[35] S. Ross, J. Pineau, S. Paquet, and B. Chaib-draa "Online planning algorithms for POMDPs," *J. Artif. Intell. Res.*, vol. 32, no. 1, pp. 663–704, Jul. 2008.
[36] M. I. Schöpe, H. Driessen, and A. Yarovoy "Optimal balancing of multi-function radar budget for multi-target tracking using lagrangian relaxation," in *Proc. 22nd Int. Conf. Inf. Fusion*, Jul. 2019.
[37] M. I. Schöpe, H. Driessen, and A. Yarovoy "Multi-task sensor resource balancing using Lagrangian relaxation and policy rollout," in *Proc. 23rd Int. Conf. Inf. Fusion*, Jul. 2020.
[38] E. Selvi, R. M. Buehrer, A. Martone, and K. Sherbondy

"Reinforcement learning for adaptable bandwidth tracking radars,"
IEEE Trans. Aerosp. Electron. Syst., vol. 56, no. 5, pp. 3904–3921, Oct. 2020.

[39] M. Shaghaghi and R. S. Adve
"Task selection and scheduling in multifunction multichannel radars,"
in *Proc. 2017 IEEE Radar Conf.*, May 2017, pp. 969–974.

[40] R. D. Smallwood and E. J. Sondik
"The optimal control of partially observable Markov processes over a finite horizon,"
Oper. Res., vol. 21, no. 5, pp. 1071–1088, 1973.

[41] M. T. J. Spaan and N. Vlassis
"Perseus: randomized point-based value iteration for POMDPs,"
J. Artif. Intell. Res., vol. 24, pp. 195–220, 2005.

[42] C. E. Thornton, M. A. Kozy, R. M. Buehrer, A. F. Martone, and K. D. Sherbondy
"Deep reinforcement learning control for radar detection and tracking in congested spectral environments,"
IEEE Trans. Cogn. Commun. Netw., vol. 6, no. 4, pp. 1335–1349, Dec. 2020.

[43] K. A. B. White, J. L. Williams, and P. Hoffensetz
"Radar sensor management for detection and tracking,"
in *Proc. 2008 11th Int. Conf. Inf. Fusion*, Jun. 2008, pp. 1–8.

[44] K. A. B. White and J. L. Williams
"Lagrangian relaxation approaches to closed loop scheduling of track updates,"
in *Proc. SPIE's Int. Symp. Opt. Sci., Eng., Instrum.*, 2012, pp. 8393–8393.

[45] J. L. Williams, J. W. Fisher, and A. S. Willsky
"Approximate Dynamic Programming for Communication-Constrained Sensor Network Management,"
IEEE Trans. Signal Process., vol. 55, no. 8, pp. 4300–4311, Aug 2007.

[46] J. Wintenby and V. Krishnamurthy
"Hierarchical resource management in adaptive airborne surveillance radars,"
IEEE Trans. Aerosp. Electron. Syst., vol. 42, no. 2, pp. 401–420, Apr. 2006.

Max Ian Schöpe received the B.Eng. degree in electrical engineering from the University of Applied Sciences in Hamburg, Germany, in February 2015. Subsequently, he started the M.Sc. program of telecommunication and sensing systems at TU Delft, the Netherlands, which he finished in October 2017. After his graduation, he started as a Ph.D. student in the Microwave Sensing, Signals and Systems group of TU Delft. His main research interests are in the areas of radar resource management using a partially observable Markov decision process framework.

Hans Driessen obtained the M.Sc. and Ph.D. degree in 1987 and 1992, respectively, both from the Department of Electrical Engineering, TU Delft. Since then he has been employed with Thales Nederland BV. He has been and is still involved with various (international) research projects and radar development programs in the area of signal/data processing and radar management. Since January 2015, he has been additionally holding a part-time position as Associate Professor at the EEMCS faculty of TU Delft in the Microwave Signals Sensor and Systems group in the field of radar systems, waveforms, and processing. His interest is in the practical application of detection, estimation, information, and control theory to various problems in sensor systems.

Alexander G. Yarovoy (FIEEE 2015) graduated from Kharkov State University, Ukraine, in 1984 with the Diploma with honors in radiophysics and electronics. He received the Candidate Phys. & Math. Sci. and Doctor Phys. & Math. Sci. degrees in radiophysics in 1987 and 1994, respectively. In 1987, he joined the Department of Radiophysics, Kharkov State University, as a Researcher and became a Full Professor there in 1997. From September 1994 through 1996, he was with the Technical University of Ilmenau, Germany, as a Visiting Researcher. Since 1999, he has been with the Delft University of Technology, the Netherlands. Since 2009, he has been leading there as Chair of Microwave Sensing, Systems and Signals. His main research interests are in high-resolution radar, microwave imaging, and applied electromagnetics (in particular, UWB antennas). He has authored and co-authored more than 450 scientific or technical papers, 6 patents, and 14 book chapters. He is the recipient of the European Microwave Week Radar Award for the paper that best advances the state-of-the-art in radar technology in 2001 (together with L. P. Ligthart and P. van Genderen) and in 2012 (together with T. Savelyev). In 2010, together with D. Caratelli, Prof. Yarovoy got the best paper award of the Applied Computational Electromagnetic Society (ACES). Prof. Yarovoy served as the General TPC chair of the 2020 European Microwave Week (EuMW'20), as the Chair and TPC Chair of the 5th European Radar Conference (EuRAD'08), as well as the Secretary of the 1st European Radar Conference (EuRAD'04). He also served as the Co-Chair and TPC Chair of the Xth International Conference on GPR (GPR 2004). He served as an Associated Editor for the *International Journal of Microwave and Wireless Technologies* from 2011 to 2018 and as a Guest Editor for five special issues of the IEEE Transactions and other journals. During the period 2008–2017, Prof. Yarovoy served as Director of the European Microwave Association (EuMA).