POMDP-Driven cognitive massive MIMO radar: joint target detection-tracking in unknown disturbances

Imad Bouhou, Stefano Fortunati, Leila Gharsalli, Alexandre Renaux

POMDP-Driven Cognitive Massive MIMO Radar: Joint Target Detection-Tracking In Unknown Disturbances

Imad Bouhou, Stefano Fortunati, Leila Gharsalli, Alexandre Renaux.

Abstract—**The joint detection and tracking of a moving target embedded in an unknown disturbance represents a key feature that motivates the development of the cognitive radar paradigm. Building upon recent advancements in robust target detection with multiple-input multiple-output (MIMO) radars, this work explores the application of a Partially Observable Markov Decision Process (POMDP) framework to enhance the tracking and detection tasks in a statistically unknown environment. In the POMDP setup, the radar system is considered as an intelligent agent that continuously senses the surrounding environment, optimizing its actions to maximize the probability of detection (P_D) and improve the target position and velocity estimation, all this while keeping a constant probability of false alarm (P_{FA}). The proposed approach employs an online algorithm that does not require any apriori knowledge of the noise statistics, and it relies on a much more general observation model than the traditional range-azimuth-elevation model employed by conventional tracking algorithms. Simulation results clearly show substantial performance improvement of the POMDP-based algorithm compared to the State-Action-Reward-State-Action (SARSA)-based one that has been recently investigated in the context of massive MIMO (MMIMO) radar systems.**

Index Terms—**Cognitive Radar, massive MIMO radars, Tracking, Partially Observable Markov Decision Process, Wald test.**

I. Introduction

Cognitive radar, first introduced by Haykin in [1], represents a major advancement in radar technology by incorporating cognitive processing capabilities. Unlike traditional systems, the cognitive radar continuously learns from its environment, dynamically optimizing its waveforms and operational parameters through real-time feedback between the transmitter and receiver. This intelligent feedback loop significantly enhances system adaptability and performance [2].

Drawing inspiration from the biological perception-action cycle, cognitive radar systems actively perceive their environment, learning crucial information about the targets and background noise. This acquired knowledge enables intelligent waveform choices, as demonstrated in [3] or adaptive

Imad Bouhou is with Université Paris-Saclay, CNRS, CentraleSupélec, Laboratoire des signaux et systèmes, 91190, Gif-sur-Yvette, France & DR2I-IPSA, 94200, Ivry-sur-Seine, France. (e-mail imad.bouhou@centralesupelec.fr).

Stefano Fortunati is with SAMOVAR, Télécom SudParis, Institut Polytechnique de Paris, 91120, Palaiseau, France (e-mail stefano.fortunati@telecom-sudparis.eu).

Leila Gharsalli is with DR2I-IPSA, 94200, Ivry-sur-Seine. (e-mail: leila.gharsalli@ipsa.fr).

Alexandre Renaux is with Université Paris-Saclay, CNRS, CentraleSupélec, Laboratoire des signaux et systèmes, 91190, Gif-sur-Yvette, France. (e-mail: alexandre.renaux@universite-paris-saclay.fr).

waveform selection using Bayesian approaches [4], resulting in a highly agile and responsive systems.

Some surveys [5], [6] have examined cognitive radar's practical applications, challenges, and efficient decision-making capabilities in dynamic environments by linking it to machine learning, optimization, and Bayesian filtering approaches to make the transmitter fully cognitive and functional.

In this work, we consider a MMIMO radar whose goal is to detect and track a target embedded in a *non-Gaussian disturbance with unknown statistical properties*. The fact that the radar is composed of a "massive" number of antenna channels enables us to achieve remarkable robustness properties that allow us to obtain performance guarantees in a much more general scenario than the standard and unrealistic Gaussian case. Specifically, as shown in [7], the massive amount of virtual antenna channels allows for the derivation of a robust Wald-type detector able to guarantee a Constant False Alarm Rate (CFAR), even when the distribution of the noise is unknown, as long as its second-order moment decays at a polynomial rate.

However, despite its notable CFAR capacity, the robust Wald-type detector cannot automatically maximize the P_D, unlike Neyman-Pearson-based detectors, since the noise statistics remain unknown. To address this crucial aspect, the authors in [8], [9] proposed a Reinforcement Learning (RL) strategy that maximizes the probability of detection P_D using the SARSA algorithm to sequentially adjust the MMIMO waveform matrix without needing apriori knowledge of the disturbance statistics. While the algorithm proposed in [8], [9] effectively detects weak targets in stationary or slow-moving scenarios, it experiences significant performance degradation in highly dynamic environments. In such cases, maximizing the P_D is insufficient, and tracking capabilities must be incorporated into the MMIMO framework.

Traditional radar tracking methods model disturbances with additive, zero-mean, Gaussian distributions (with possibly unknown covariance matrix) and use the traditional range-azimuth-elevation model as an observation model. This conventional framework enables the application of Bayesian filtering techniques, which were previously employed by cognitive radar frameworks in [1], [10], [11] to estimate the current environmental state and predict its evolution.

To overcome these two critical limitations, i.e., *i)* the need for apriori knowledge of the disturbance statistics and *ii)* the stationarity of the targets, this work proposes a cognitive radar framework that is based on a MMIMO radar system

able to merge the robustness/CFAR property of the Wald-type detector [7] with the ability of Reinforcement Learning (RL)-based algorithm to maximize the P_D while enhancing tracking performance. Specifically, to address the possible non-stationarity of the scenario, the joint detection-tracking problem is modeled as a Partially Observable Markov Decision Process (POMDP) [12]. In simple terms, the "observable part" of the process is the detection made by the robust Wald-type detector and the inherent estimation of the target power. However, the "unobservable part" — that must instead be deduced from the observations — is the target's actual state, specifically, its Cartesian coordinates in a given reference frame and the relevant velocity vector.

POMDPs, as opposed to a Markov Decision Process (MDP), can handle sequential decision-making when the states are considered to be hidden. The agent when handling POMDPs, has to make actions based on the history of all the observations it received in the past. POMDPs are hard to solve, so approximation approaches are used, such as Deep recurrent Q-learning networks (DRQN) [13] or online-learning algorithms that are based on the UCT algorithm [14] like Partially Observable Monte-Carlo Planing (POMCP) [15].

In radar applications, some previous works have adopted the POMDP design to track targets. For instance, in [16], the authors proposed a DRQN equipped with an attention-head [17] in addition to a long-short term memory (LSTM) to be able to track a changing number of targets. In [18], [19], the authors applied Deep Q-learning Networks (DQN) [20] and managed to improve tracking precision. However, both DQN and DRQN frameworks are affected by the same limitation: they require a representative dataset of the environment. This may be an unrealistic requirement since it would be difficult to collect a training dataset that contains examples for all the potential scenarios, given all the possible disturbance distributions and all the potential target trajectories. For this reason, it is preferable to use online algorithms that plan and adapt themselves sequentially to the information gathered in real-time.

Some previous works have already exploited online POMDP solvers to handle target tracking in radar applications; for instance, in [21], the authors showed that the use of Monte Carlo Tree Search with double progressive widening (MCTS-DPW) [22] improves tracking performance. A similar approach was proposed in [23] but with more guidance to how the tree search explores the action space, and this is done using Voronoi progressive widening (MCTS-VPW)[24].

In this work, we propose an original approach able to merge the POMCP machinery with the statistical robustness of the robust Wald-detector [7] in order to come up with an algorithm providing nearly optimal tracking performance in the presence of *unknown disturbance statistics*. The "disturbance agnostic" capacity represents the crucial advantage of our proposed method with respect to competing algorithms, e.g., [23] and [21], that require the apriori knowledge of the disturbance distribution. Further details on this fundamental point are detailed in II-C.

Notations: In this paper, matrices are denoted by uppercase letters $\mathbf{A}$ and vectors by lowercase letters $\mathbf{a}$. $(\cdot)^T$, $(\cdot)^H$, and $(\cdot)^*$ represent transpose, conjugate transpose, and complex conjugate, respectively. $\mathbf{I}_N$ is the $N \times N$ identity matrix and $\mathbf{0}_N$ is a zero vector of size N. $\otimes$ denotes Kronecker product. A closed intervals between a and b is denoted as $[a, b]$, and sets as $\{a, b\}$. $|\cdot|$ is the absolute value. For probability distributions, the chi-squared distribution with k degrees of freedom is denoted as $\chi_k^2(\delta)$, where δ is the non-centrality parameter (then the distribution is central if $\delta = 0$). The real Gaussian distribution with mean μ and a variance σ^2 is represented as $\mathcal{N}(\mu, \sigma^2)$ (and $\mathcal{CN}(\mu, \sigma^2)$ when it is a circular complex Gaussian). Finally, the exponential distribution with a rate parameter x is written as $\text{Exp}(x)$.

II. PROBLEM FORMULATION

This section presents the radar system model, defines mathematically what is a POMDP, and summarizes the POMCP algorithm employed in this paper.

A. System Model

The MMIMO signal model considered here is the same as the one used in [7] and [8]. To avoid redundancy, in the following, we recall only the main points that are needed for the subsequent derivations while we refer to [7] and [8] for an in-depth description.

We consider a co-located MIMO radar equipped with N_T transmit and N_R receive physical antennas. Following the standard MIMO theory, we indicate as $N = N_T N_R$ the total number of *virtual spatial antenna channels*. The radar field of view is assumed to be discretized into L angle bins $\{\theta_l; l = 1, .., L\}$, and in total the system transmits $T_{\max}$ scans, $t \in \{0, .., T_{\max} - 1\}$. Therefore, the detection problem can be re-formulated for an angle bin l and a time step t as [8], [7]:

$$
\begin{aligned}
H_0 &: \mathbf{y}_{t+1,l} = \mathbf{c}_{t+1,l}, \\
H_1 &: \mathbf{y}_{t+1,l} = \alpha_{t+1,l} \mathbf{v}_{t,l} + \mathbf{c}_{t+1,l},
\end{aligned}
\tag{1}
$$

where $\mathbf{c}_{t+1,l} \in \mathbb{C}^N$ is a random vector of unknown probability density function p_C, representing the disturbance. We assume that the N entries of the disturbance vector are sampled from a circular complex random process $\{c_{t+1,l,n}, \forall n\}$. We only assume that its auto-correlation function exists and decays *at least* at a polynomial rate [7, Assumption 1]. The unknown deterministic scalar $\alpha_{t+1,l} \in \mathbb{C}$ stands for the radar-cross section and the two-way path loss. The known vector $\mathbf{v}_{t,l} \in \mathbb{C}^N$ is defined as in [7] and [8]:

$$
\mathbf{v}_{t,l} = (\mathbf{W}_t^T \mathbf{a}_T(\theta_l)) \otimes \mathbf{a}_R(\theta_l),
\tag{2}
$$

where $\mathbf{W}_t \in \mathbb{C}^{N_T \times N_T}$ is the waveform matrix and $\mathbf{a}_T(\theta_l)$ and $\mathbf{a}_R(\theta_l)$ are the transmit and receive steering vectors, assumed to be perfectly known. The waveform matrix $\mathbf{W}_t$ that focuses the transmitted energy on angle bin θ_l is determined by solving the following optimization problem [25], [26]:

$$
\begin{aligned}
\max_{\mathbf{W}} \quad & \mathbf{a}_T^T(\theta_l) \mathbf{W} \mathbf{W}^H \mathbf{a}_T^*(\theta_l) \\
\text{s.t.} \quad & \text{Tr}(\mathbf{W} \mathbf{W}^H) = P_T,
\end{aligned}
\tag{3}
$$

where P_T is the total transmit power of the radar.
Clearly, the waveform matrix $\mathbf{W}_t$ is a solution to (3) if and only if it is a square-root of the matrix $\frac{P_T}{N_T} \mathbf{a}_T^*(\theta_l) \mathbf{a}_T^T(\theta_l)$.

As shown in [7], the hypothesis testing problem in (1) can be solved using the following robust Wald-type test:

$$\Lambda_{t+1,l} = 2|\hat{\alpha}_{t+1,l}|^2 \frac{||\mathbf{v}_{t,l}||^4}{\mathbf{v}_{t,l}^H \widehat{\boldsymbol{\Sigma}}_{t+1,l} \mathbf{v}_{t,l}} \underset{H_0}{\overset{H_1}{\gtrless}} \lambda, \qquad (4)$$

where $\widehat{\boldsymbol{\Sigma}}_{t+1,l}$ is an estimator of the disturbance covariance matrix that can be computed the same way as in [7], [27], while $\hat{\alpha}_{t+1,l} = (\mathbf{v}_{t,l}^H \mathbf{y}_{t+1,l})/||\mathbf{v}_{t,l}||^2$ is an estimator of the parameter $\alpha_{t+1,l}$. The threshold λ is chosen to keep the probability of false alarm below a predetermined value, say P_{FA}, by solving the following equation:

$$P_{FA} = \Pr\{\Lambda_{t+1,l} > \lambda | H_0\}. \qquad (5)$$

In [7], the authors proved that $\Lambda_{t+1,l}|H_0 \underset{N \to \infty}{\sim} \chi_2^2(0)$. Therefore, (5) becomes:

$$P_{FA} \underset{N \to +\infty}{=} \exp(-\lambda/2). \qquad (6)$$

The threshold λ is computed by choosing a desired P_{FA} using $\lambda = -2\log(P_{FA})$.

The probability of detection P_D is defined by the equation:

$$P_D = \Pr\{\Lambda_{t+1,l} > \lambda | H_1\}. \qquad (7)$$

In [7], the authors proved that $\Lambda_{t+1,l}|H_1 \underset{N \to \infty}{\sim} \chi_2^2(\zeta_{t+1,l})$ where $\zeta_{t+1,l} = 2|\alpha_{t+1,l}|^2 ||\mathbf{v}_{t,l}||^4/(\mathbf{v}_{t,l}^H \boldsymbol{\Sigma} \mathbf{v}_{t,l})$ and $\boldsymbol{\Sigma}$ is the true unknown disturbance covariance. Therefore, (7) becomes:

$$P_D \underset{N \to +\infty}{=} Q_1(\sqrt{\zeta_{t+1,l}}, \sqrt{\lambda}), \qquad (8)$$

where Q_1 is the Marcum Q function of order 1.

Given that the probability density function of the disturbance p_C is unknown, $\zeta_{t+1,l}$ needs to be estimated by $\Lambda_{t+1,l}$. Hence, the probability of detection can be approximated by:

$$P_D \underset{N \to +\infty}{\simeq} Q_1 \left(\sqrt{2|\hat{\alpha}_{t+1,l}|^2 \frac{||\mathbf{v}_{t,l}||^4}{\mathbf{v}_{t,l}^H \widehat{\boldsymbol{\Sigma}}_{t+1,l} \mathbf{v}_{t,l}}}, \sqrt{\lambda} \right). \qquad (9)$$

The waveform matrix $\mathbf{W}_t$ has to be chosen to maximize the probability of detection P_D.

B. Reinforcement learning (POMDPs): a recall

Reinforcement learning (RL) is a branch of machine learning in which an agent learns to make decisions by interacting with an environment [28]. The main goal is to maximize a cumulative reward over time. In this framework, the agent explores various actions, observes their consequences, and adjusts its strategy accordingly.

A POMDP [12] can be described by the tuple $(\mathcal{S}, \mathcal{O}, \mathcal{A}, \mathcal{P}, \Omega, \mathcal{R})$, where $\mathcal{S}$ represents the set of states, $\mathcal{O}$ represents the set of observations, $\mathcal{A}$ the set of available actions. The state transition probabilities

$$\mathcal{P}_{s,s'}^a = p(S_{t+1} = s' | S_t = s, A_t = a), \qquad (10)$$

define the likelihood of transitioning from the current state s to a new state s' after taking action a. Similarly, the observation probabilities

$$\Omega_{s',o}^a = p(O_{t+1} = o | S_{t+1} = s', A_t = a), \qquad (11)$$

specify the likelihood of observing o after executing action a and arriving at state s'. Finally, $\mathcal{R}_{s,s'}^a$ is the reward obtained after executing action a, being in a state s and reaching state s'. Additionally, one needs to define other quantities useful for the following. For instance, a history h_t is defined by:

$$h_t = (a_0, o_1, ..., a_{t-1}, o_t), \qquad (12)$$

which is a sequence of the previously taken actions and the obtained observations. The agent's goal is to develop a policy

$$\pi(a|h) = p(A_t = a | H_t = h), \qquad (13)$$

that maps histories to a probability distribution over actions. In addition to the history, the agent can also build a posterior distribution:

$$b(s|h) = p(S_t = s | H_t = h), \qquad (14)$$

called the belief state.

In the POMDP, the value function $V_\pi(h)$ is the expected return starting from a history h and following a policy π,

$$V_\pi(h) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}_{s_t, s_{t+1}}^{a_t} \left| \begin{array}{l} h_0 = h; s_0 \sim b(.|h_0) \\ t \geq 0 : a_t \sim \pi(.|h_t) \\ t \geq 0 : s_{t+1} \sim \mathcal{P}_{s_t}^{a_t} \\ t \geq 0 : o_{t+1} \sim \Omega_{s_{t+1}}^{a_t} \\ h_{t+1} = (h_t, a_t, o_{t+1}) \end{array} \right. \right], \qquad (15)$$

where $\gamma \in (0,1)$ is called the discount factor. The optimal value function $V^*(h) = \max_\pi V(h)$ is the value function computed with the optimal policy π^*.

When the agent has executed an action a and observed an observation o, it has to update the belief state with the new information (a, o) using the following equation:

$$b(s'|h, a, o) \propto \Omega_{s',o}^a \sum_s \mathcal{P}_{s,s'}^a b(s|h). \qquad (16)$$

C. POMCP algorithm

The POMCP algorithm [15] is an online planning algorithm in large POMDPs when both the actions and observations are discrete. It is an extension of the UCT algorithm [14] to POMDPs by building a MDP whose states are defined by histories.

The POMCP needs a black-box generator $\mathcal{G}(s, a) = (s', o, r)$ where $r = \mathcal{R}_{s,s'}^a$, instead of knowing the distributions $\mathcal{P}$ and Ω explicitly. The POMCP tree is initialized at the root with a belief set B, an estimation of the belief state $b(s|h)$, containing N_p unweighted particles.

In Fig. 1, each red node in the tree contains $(V(h), N(h), B(h))$, representing the value of the history h, the number of times h has been visited, and the belief set, respectively. The red nodes $(V(h), N(h), B(h))$ have blue children nodes defined by $(Q(h, a), N(h, a))$ for all possible actions $a \in \mathcal{A}$. Here, $N(h, a)$ is the number of times where action a was taken in history h, and $Q(h, a)$ is the value of being in history h followed by the execution of the action a. Blue nodes generate an observation o, leading to new red child nodes. These new nodes are characterized by an updated history combining the previous history h with the new information pair (a, o).

Fig. 1: POMCP Tree illustration with two actions and two observations.

Algorithm 1 Simulate

1: **procedure** SIMULATE(s, h, $depth$)
2:     **if** $\gamma^{depth} < \varepsilon$ **then**
3:         **return** 0
4:     **end if**
5:     $a \leftarrow \arg\max_b Q(h,b) + c\sqrt{\frac{\log N(h)}{N(h,b)}}$
6:     $(s',o,r) \sim \mathcal{G}(s,a)$
7:     **if** $\texttt{Node}(h,a,o) \notin \texttt{Tree}$ **then**
8:         Add $\texttt{Node}(h,a,o)$
9:         **return** ROLLOUT$(s',h,depth)$
10:     **end if**
11:     $R \leftarrow r + \gamma \cdot$ SIMULATE$(s',hao,depth+1)$
12:     **if** $depth \neq 0$ **then**
13:         $B(h) \leftarrow B(h) \cup \{s\}$
14:     **end if**
15:     $N(h) \leftarrow N(h) + 1$
16:     $N(h,a) \leftarrow N(h,a) + 1$
17:     $Q(h,a) \leftarrow Q(h,a) + \frac{R-Q(h,a)}{N(h,a)}$
18:     **return** R
19: **end procedure**
20: **procedure** ROLLOUT(s, h, $depth$)
21:     **if** $\gamma^{depth} < \varepsilon$ **then**
22:         **return** 0
23:     **end if**
24:     $a \sim \pi_{\text{rollout}}(.|h)$
25:     $(s',o,r) \sim \mathcal{G}(s,a)$
26:     **return** $r + \gamma$ROLLOUT$(s',hao,depth+1)$
27: **end procedure**

Algorithm 2 Solve

Require: N_{sim} ▷ Number of simulations.
Require: B ▷ Belief set at the root.
1: **procedure** POMCP(h)
2:     **for** each simulation $i = 1,..,N_{\text{sim}}$ **do**
3:         $s \sim B$
4:         SIMULATE$(s,h,0)$
5:     **end for**
6:     **return** $\arg\max_a Q(h,a)$
7: **end procedure**

The agent runs N_{sim} simulations through the tree search and computes the value of the current history $V(h)$ at the root of the tree. Algorithm 1 presents the simulation process. It starts by sampling a state from the belief set B at the root of the tree and then selecting the actions that maximize the Upper Confidence Bound (UCB1) [29] criterion given the actions of the child nodes $Q^{UCT}(h,a) = Q(h,a) + c\sqrt{\frac{\log(N(h))}{N(h,a)}}$ where c is a hyperparameter set to balance between exploration and exploitation. When a leaf is encountered, a new node is added, from which the rollout stage begins, which consists of running a simulation starting from the newly added node using some rollout policy π_{rollout}. To run the simulation through the tree, the algorithm uses a black-box generator $\mathcal{G}(s,a)$.

As stated in Theorem 1 in [15], as the number of simulations increases, the value function $V(h)$ computed by the POMCP at the root of the tree converges in probability to $V^*(h)$.
At the end of the simulations, the agent chooses the action $a^* = \arg\max_{a \in \mathcal{A}} Q(h,a)$, as shown in Algorithm 2.
After executing the optimal action and observing o, the belief update proceeds as follows: the algorithm selects a random particle s from B, generates (s',o',r) using $\mathcal{G}(s,a)$, and compares o' to o. If $o' = o$, then s' is added to the new belief set B'. This process is repeated until the new belief B' contains N_p particles.

III. COGNITIVE RADAR AS A POMDP

In this section, we explain how the aforementioned POMDP definitions can be fitted into the radar framework and provide a full algorithm description. In this work, the algorithm is used with the assumption of having only one single target in the environment.

A. Action space

The action is related to the set of possible waveform matrices from which the radar can choose. The radar's angular resolution is represented by L angle bins, so the radar will have L actions, which means the radar can choose one waveform matrix out of L that allows it to focus all the energy in a single angle bin. At time step t, the radar simply chooses an angle bin $l \in \{1, 2...L\}$ associated with an angle θ_l, and the waveform matrix $\mathbf{W}_t$ is computed as the solution of the optimization problem (3), i.e., the matrix $\mathbf{W}_t$ is the square root of the matrix $\frac{P_T}{N_T}\mathbf{a}_T^*(\theta_l)\mathbf{a}_T^T(\theta_l)$.

B. State space

The state space consists of the target's possible positions and velocities. The state space consists of the target's possible positions and velocities. At time step t, the state is defined as $\mathbf{s}_t = [x_t, V_{x,t}, y_t, V_{y,t}]^T$ where $[x_t, y_t]$ and $[V_{x,t}, V_{y,t}]$ denote

the target position and velocity vectors respectively.
The following equation describes the dynamics of the target:

$$\mathbf{s}_{t+1} = \mathbf{A}\mathbf{s}_t + \mathbf{G}\mathbf{w}_t, \tag{17}$$

where $\mathbf{A}$ is the state transition block-matrix :

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_b & \mathbf{0}_{2\times 2} \\ \mathbf{0}_{2\times 2} & \mathbf{A}_b \end{bmatrix}, \mathbf{A}_b = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}. \tag{18}$$

The term $\mathbf{G}\mathbf{w}_t$ represents the noise, and the matrix $\mathbf{G}$ can also be written in block form as:

$$\mathbf{G} = \begin{bmatrix} \mathbf{G}_b & \mathbf{0}_{2\times 1} \\ \mathbf{0}_{2\times 1} & \mathbf{G}_b \end{bmatrix}, \mathbf{G}_b = \begin{bmatrix} \Delta t^2/2 \\ \Delta t \end{bmatrix}. \tag{19}$$

$$\mathbf{w}_t \sim \mathcal{N}\left(\mathbf{0}_2, \sigma_s^2 \mathbf{I}_2\right), \tag{20}$$

where σ_s is the standard deviation of the process noise.
Given (17), the future state only depends on the current state, i.e., in this case, the transition probability does not depend on the action taken by the radar.

C. Observation space

At time step t, the radar chooses an action a associated with an angle bin l from which a vector $\mathbf{v}_{t,l}$ is computed, then the radar observes the estimation of the parameter $|\alpha_{t+1,l}|$ when there is a detection.

$$o_{t+1} = \begin{cases} |\hat{\alpha}_{t+1,l}| & \text{if } \Lambda_{t+1,l} \geq \lambda, \\ \emptyset & \text{otherwise,} \end{cases} \tag{21}$$

In accordance with the classical "radar equation" theory, the parameter $|\alpha_{t+1,l}|$ exhibits an inverse proportional relationship with R_{t+1}^2 [30], where R_{t+1} represents the range between the target and radar.

$$|\alpha_{t+1,l}| \propto 1/R_{t+1}^2. \tag{22}$$

In [7], it has been shown that the estimated parameter $\hat{\alpha}_{t+1,l}$ is asymptotically distributed as:

$$\frac{\hat{\alpha}_{t+1,l} - \alpha_{t+1,l}}{\hat{\sigma}_{t,l}} \underset{N\to\infty}{\sim} \mathcal{CN}(0,1), \tag{23}$$

where $\hat{\sigma}_{t,l} = \sqrt{\mathbf{v}_{t,l}^H \hat{\mathbf{\Sigma}}_{t+1,l} \mathbf{v}_{t,l}}/||\mathbf{v}_{t,l}||^2$ is given in [7].
The POMCP is an online POMDP solver when the observations are discrete. Since, in our case, according to (21), the observation space is continuous, it must be transformed into a discrete one by choosing a discretization step β_l.
The result in (23) allows to approximate the distribution of $|\hat{\alpha}_{t+1,l} - \alpha_{t+1,l}|^2$ with the exponential distribution $\text{Exp}(\hat{\sigma}_{t,l}^2)$ when N goes to infinity. Then, we define the discretization step β_l as the real number verifying the following condition:

$$\Pr\{|\,|\hat{\alpha}_{t+1,l}| - |\alpha_{t+1,l}|\,| < \beta_l\} \geq 0.95. \tag{24}$$

It is enough that β_l verifies the following condition:

$$\Pr\{|\,\hat{\alpha}_{t+1,l} - \alpha_{t+1,l}\,|^2 < \beta_l^2\} = 0.95, \tag{25}$$

to also verify the condition (24). Therefore, using the analytic form of the cumulative distribution function of the exponential distribution $\text{Exp}(\hat{\sigma}_{t,l}^2)$, one gets $\beta_l = \sqrt{3}\hat{\sigma}_{t,l}$.

D. Reward function

The reward function should encourage the radar to detect and track the target in the environment. In the POMDP definition, the reward function depends on the current state $\mathbf{s}$, the taken action a, and the next state $\mathbf{s}'$. The action a is about choosing an angle bin θ_a where the target will be located in the future.
Let us indicate by $\theta_{\mathbf{s}'}$ the true future angle bin of the target. To encourage accurate target position prediction, the reward function is chosen as:

$$\mathcal{R}_{s,s'}^a = \mathbf{1}\{\theta_a = \theta_{\mathbf{s}'}\}. \tag{26}$$

Note that the reward function here does not depend on the current state $\mathbf{s}$.

E. Simulation model

As previously stated, the POMCP algorithm [15] needs a black-box generator $\mathcal{G}(s,a) = (s', o, r)$ to be able to run simulations through the tree search.
In the context of this paper, the noise disturbance p_C is assumed to be unknown, which makes the observation probabilities unknown and consequently makes the POMCP impossible to use.
Fortunately, using the asymptotic distribution of the estimated parameter $\hat{\alpha}_{t+1,l}$, a generator can be put in place to be used by the POMCP and the particle filter to maintain the detection and tracking of a target in the environment.

Algorithm 3 Generator $\mathcal{G}(\mathbf{s}_t, a_t)$.

Require: $\mathbf{s}_t = (x_t, V_{x,t}, y_t, V_{y,t})^T$ and action a_t.
Require: $(\hat{\sigma}_l)_{l\in\{1,..,L\}}$.
1: $\mathbf{s}_{t+1} \leftarrow \mathbf{A}\mathbf{s}_t + \mathbf{G}\mathbf{w}_t$
2: $\theta_{t+1} \leftarrow \text{GetAngleBin}(\mathbf{s}_{t+1})$
3: $l_t \leftarrow \text{GetAngleBin}(a_t)$
4: $\alpha_{t+1} \leftarrow \text{GetRCS}(\mathbf{s}_{t+1})$ ▷ (22)
5: $\hat{\alpha}_{t+1} \leftarrow \mathcal{CN}(\alpha_{t+1}, \hat{\sigma}_{l_t}^2)$ ▷ (23)
6: $\Lambda_t \leftarrow 2|\hat{\alpha}_{t+1}|^2/\hat{\sigma}_{l_t}^2$ ▷ (27)
7: **if** $l_t \neq \theta_{t+1}$ **then** $o_{t+1} \leftarrow \emptyset$
8: **else if** $l_t = \theta_{t+1}$ **then**
9:      **if** $\Lambda_t \geq \lambda$ **then** $o_{t+1} \leftarrow |\hat{\alpha}_{t+1}|$
10:      **else if** $\Lambda_t < \lambda$ **then**
11:          $o_{t+1} \leftarrow \emptyset$
12:      **end if**
13: **end if**
14: $r_t \leftarrow \mathbf{1}\{l_t = \theta_{t+1}\}$
15: **return** (s_{t+1}, o_{t+1}, r_t)

In the Algorithm 3, the `GetAngleBin` function determines the angle bin based on the target's coordinates or the radar's action. The `GetRCS` function computes the parameter $\alpha_{t+1,l} = |\alpha_{t+1,l}|e^{j\phi}$, where ϕ is uniformly sampled from $(0, 2\pi)$. According to (22), the magnitude $|\alpha_{t+1,l}|$ depends on the target's position.

This proposed generator requires parameters $(\hat{\sigma}_l)_{l\in\{1,..,L\}}$ to be used in the asymptotic relation (23). To simulate the

detection, the decision statistic can be reformulated using $\widehat{\sigma}_{t,l}^2$ to obtain the following equation:

$$\Lambda_{t+1,l} = 2|\hat{\alpha}_{t+1,l}|^2 / \widehat{\sigma}_{t,l}^2. \tag{27}$$

The decision statistic inside the generator will take the same format as in (27) by replacing $\widehat{\sigma}_{t,l}$ with $\widehat{\sigma}_l$.

The generator will be used by the POMCP to run simulations in the tree search and will also be used by the unweighted particle filter to make predictions and fill up the belief set.

IV. Cognitive Radar Design

This section introduces an unweighted particle filter capable of handling distribution agnostic observation models and then explains how the POMCP is incorporated into the cognitive radar framework.

Unlike standard tracking tasks with known observation models, our scenario involves an unknown observation model due to the unspecified disturbance p_C. To address this, the typical particle filter is replaced with an unweighted version. Additionally, note that it is employed with a non-standard observation model, departing from the conventional range-azimuth-elevation format used in most tracking applications.

A. The particle filter

The particle filter presented in this section is exactly the same as the one used by the POMCP to estimate the posterior. Here, instead of running simulations inside the tree search, this approach only makes a prediction about the target's position, and given that prediction, an action is selected.

At time step t, an approximation of $b(.|h_t)$ is defined by the set B_t. Similarly to the POMCP, which is driven with rewards, the particle filter needs to predict the future hidden state of the target, hence, compute $\mathbb{E}(\mathbf{s}_{t+1}|h_t)$.

By definition of the target dynamics, the transition probabilities only depend on the previous state, therefore, $\mathcal{P}_{\mathbf{s},\mathbf{s}'}^a = \mathcal{P}_{\mathbf{s},\mathbf{s}'}$.

$$\mathbb{E}(\mathbf{s}_{t+1}|h_t) = \int_{\mathbf{s}'} \mathbf{s}' p(\mathbf{s}_{t+1} = \mathbf{s}'|H_t = h_t) d\mathbf{s}' \tag{28}$$

$$= \int_{\mathbf{s}'} \mathbf{s}' \int_{\mathbf{s}} \mathcal{P}_{\mathbf{s},\mathbf{s}'} b(\mathbf{s}|h_t) d\mathbf{s} d\mathbf{s}' \tag{29}$$

$$= \int_{s} \mathbb{E}(\mathbf{s}_{t+1}|\mathbf{s}_t = \mathbf{s}) b(\mathbf{s}|h_t) d\mathbf{s} \tag{30}$$

$$= \mathbb{E}_{\mathbf{s}_t \sim b(.|h_t)} \left[\mathbb{E}(\mathbf{s}_{t+1}|\mathbf{s}_t) \right] \tag{31}$$

$$\approx \frac{1}{|B_t|} \sum_{\mathbf{s} \in B_t} \mathbb{E}(\mathbf{s}_{t+1}|\mathbf{s}_t = \mathbf{s}) \tag{32}$$

In essence, when the particle filter computes the prediction, it predicts the angle bin where the target will be in the future, therefore it makes sense to select the action related to the predicted angle bin.

B. POMCP in the cognitive radar setup

The radar chooses the orthogonal waveform matrix $\mathbf{W}_{\text{orth}} = \sqrt{\frac{P_T}{N_T}} \mathbf{I}_{N_T}$ at the beginning until there is a detection, then it uses the observations to estimate the coordinates of the target and initializes the velocities uniformly using an interval

$[-V_{\max}, V_{\max}]$, where $V_{\max}$ is some maximum velocity. During this phase, the standard deviations $(\widehat{\sigma}_l)_{l \in \{1,..,L\}}$ to use in the asymptotic relation (23) can be computed and saved to be used later in the generator.

In the Algorithm 4, the full cognitive radar design is presented, and, particularly, how the POMCP is integrated into it.

Algorithm 4 Cognitive radar design.

Require: N_{sim} ▷ Number of simulations.
Require: B_0 ▷ Initial Belief set.
Require: $\mathcal{G}$ ▷ A black-box generator.
Require: $(\widehat{\sigma}_l)_{l \in \{1,..,L\}}$ ▷ Parameters for the generator.
Require: $\beta_l = \sqrt{3}\widehat{\sigma}_l \quad l = 1, \cdots, L$ ▷ Parameters for the generator.
1: **for** each time step $t = 0, .., T_{\max} - 1$ **do**
2:    $a_t \leftarrow$ POMCP.Solve(N_{sim}, B_t).   ▷ Algorithm 2
3:    $l \leftarrow$ GetAngleBin(a_t).
4:    Compute $\mathbf{W}_t$ from l.
5:    Receive the signal $\mathbf{y}_{t+1,l}$ and compute observations.
6:    Observe o_{t+1} from (21).
7:    $B_{t+1} \leftarrow$ UpdateBelief(B_t, a_t, o_{t+1}).
8: **end for**

V. Simulations

The disturbance model used here will be the same as in [8] and is based on an auto-regressive process (AR) of order p.

$$c_n = \sum_{i=1}^{p} \rho_i c_{n-i} + w_n, \quad n \in (-\infty, +\infty), \tag{33}$$

The process is driven by identically independent distributed i.i.d. t-distributed innovations w_n with a probability density function p_w defined as:

$$p_w(w_n) = \frac{\mu}{\sigma_w^2 \pi} \left(\frac{\mu}{\xi} \right)^{\mu} \left(\frac{\mu}{\xi} + \frac{|w_n|^2}{\sigma_w^2} \right)^{-(\mu+1)}, \tag{34}$$

where $\mu \in (1, +\infty)$ is the shape parameter controlling the non-Gaussianity of w_n and the scale parameter is defined as $\xi = \frac{\mu}{\sigma_w^2(\mu-1)}$. In the simulations, the parameters used are $p = 6$ for the order of the AR process, $\mu = 2$, $\sigma_w^2 = 1$ and the coefficient vector ρ is defined as:

$$\rho = [0.5e^{-j2\pi \cdot 0.4}, 0.6e^{-j2\pi \cdot 0.2}, 0.7e^{-j2\pi \cdot 0},$$
$$0.4e^{-j2\pi \cdot 0.1}, 0.5e^{-j2\pi \cdot 0.3}, 0.6e^{-j2\pi \cdot 0.35}]^T. \tag{35}$$

The number of spatial channels $N = N_T N_R = 10^4$, the number of angle bins $L = N_T = 100$, the total power $P_T = 1$ and the probability of false alarm $P_{FA} = 10^{-4}$.

The table I contains the parameter settings of the POMCP algorithm.

TABLE I: POMCP parameters

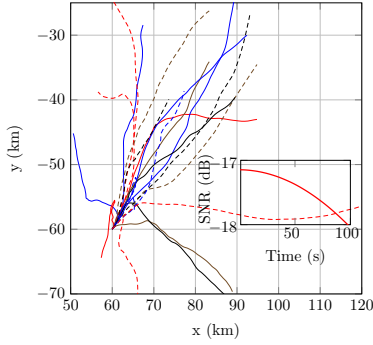| Parameter	Value	Parameter	Value
Number of simulations N_{sim}	10^4	Discount factor γ	0.8
Number of particles N_p	10^4	Time steps	100
UCB1 parameter c	$\sqrt{2}$	Tree Depth	2

Fig. 2: Study case 1: Potential trajectories with $\mathbf{s}_0 = (60\text{km}, 0.2\text{km/s}, -60\text{km}, 0.2\text{km/s})^T$ and noise $\sigma_s = 0.03$. The inner figure represents the average of SNR trajectories.
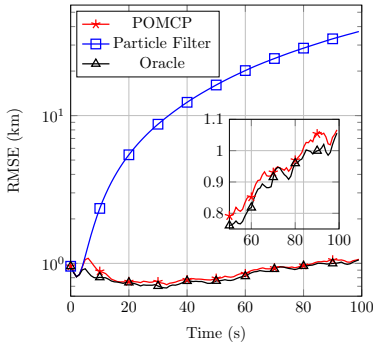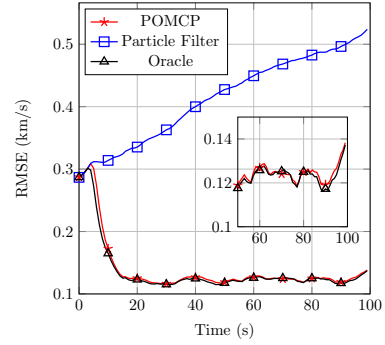

Fig. 4: Study case 1: RMSE between the estimated and true velocities of the target for each algorithm.

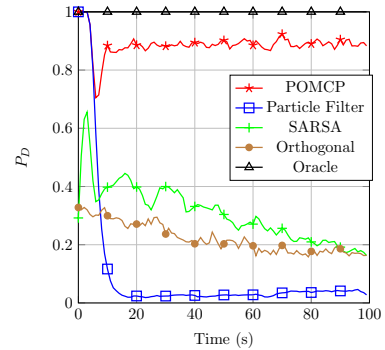Fig. 3: Study case 1: RMSE between the estimated and true coordinates of the target for each algorithm.

Fig. 5: Study case 1: Probability of detecting a moving target for different approaches.

A step size satisfying the condition (24) is chosen as $\beta_l = \sqrt{3}\widehat{\sigma}_l$ for all $l \in 1, \ldots, L$. Results are averaged over 250 Monte Carlo simulations.

To establish a lower bound for performance evaluation, we define an Oracle algorithm. The Oracle is an idealized algorithm that knows the future angle bin containing the target (but not its exact coordinates or velocity) and always takes the optimal action based on this knowledge. It receives an observation and builds a belief state. In essence, the oracle is the equivalent of a flawless POMCP.

The POMCP, particle filter, and Oracle are compared for the tracking task using the root-mean-square error (RMSE) for coordinate and velocity estimation. For the detection task, these three algorithms, along with the SARSA algorithm and the orthogonal waveform, are evaluated based on the probability of detection P_D.

A. Study Case 1: Slow target

In this scenario, one target is considered and initialized with $\mathbf{s}_0 = (60\text{km}, 0.2\text{km/s}, -60\text{km}, 0.2\text{km/s})^T$, with noise $\sigma_s = 0.03$. Figure 2 shows potential trajectories and the average SNR trajectories, which start at -17dB and decrease to -18dB after 100-time steps, i.e., most SNR trajectories decrease.

Figure 3 shows an increasing RMSE for the particle filter, which can be explained by the fact that the target becomes lost early on, which is confirmed in Figure 5. However, the POMCP estimates stay close to Oracle ones.

Figure 4 shows a bad velocity estimation at the start of the simulation but improves as the POMCP gathers more observations, which leads to the filtration of particles that do not represent the hidden state.

As shown in Figure 5, the particle filter loses the target as it moves, eventually leading to bad position and velocity estimations. It also shows a robust performance of the POMCP in keeping the probability of detection high, significantly outperforming the SARSA and the particle filter, which drops below 0.2. Given that the SNR is low, the orthogonal waveform does not have a high probability of detection.

B. Study Case 2: Fast target

In this scenario, one target is considered and initialized with $\mathbf{s}_0 = (60\text{km}, 0.35\text{km/s}, -60\text{km}, 0.35\text{km/s})^T$ and $\sigma_s = 0.005$. Figure 6 shows that the target has linear trajectories given its high speed. In the inner plot, the SNR trajectories, on average, start from -17dB and decrease to -20dB after 100-time steps.

Figures 9, 8 and 7 present similar patterns to the previous study case. The RMSE values for POMCP estimates demonstrate smaller deviations from the Oracle ones than the particle filter estimates. The detection probabilities in Figure 9 indicate that target detection with the POMCP remains feasible under these conditions.

Given that the SNR in this case is much lower than the previous one, it is normal that the probability of detection
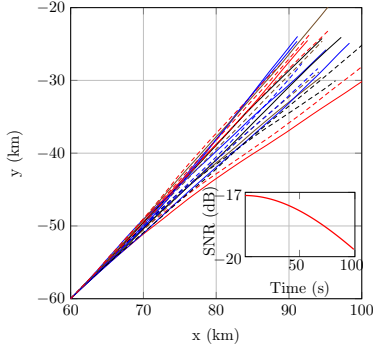
Fig. 6: Study case 2: Potential trajectories with $\mathbf{s}_0 = (60\text{km}, 0.35\text{km/s}, -60\text{km}, 0.35\text{km/s})^T$ and noise $\sigma_s = 0.005$. The inner figure represents the average of SNR trajectories.
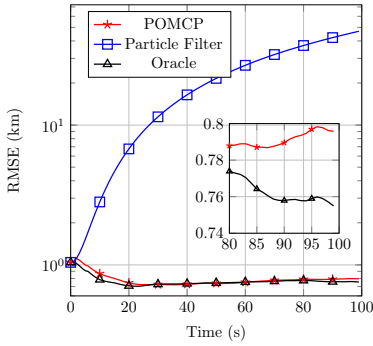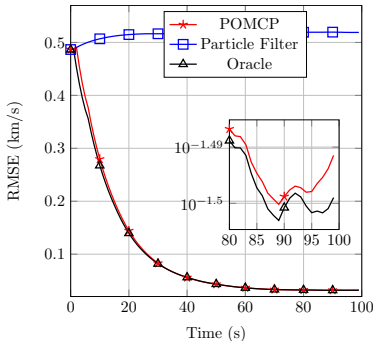

Fig. 7: Study case 2: RMSE between the estimated and true coordinates of the target for each algorithm. The inset highlights the close behavior of the POMCP and Oracle curves.

with the orthogonal waveform is low. The POMCP manages to detect a fast moving target and maintains a probability of detection above 0.8.

C. Tuning the depth of the tree in POMCP

In the previous study cases, the depth of the tree was chosen to be equal to 2 for two main reasons:

- **Short-term planning**: A shallow tree depth (2–3 steps) is sufficient to optimize actions for the short-term goal of estimating the hidden state at each time step.

Fig. 8: Study case 2: RMSE between the estimated and true velocities of the target for each algorithm.
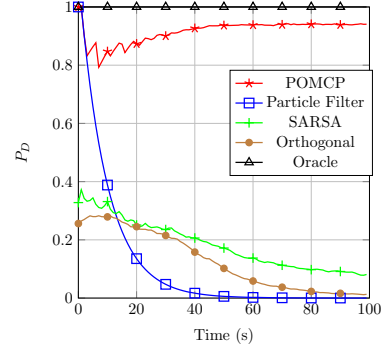
Fig. 9: Study case 2: Probability of detecting a moving target for different approaches.

- **Cancel the generator's bias:** Increasing the tree depth would cause the tree's beliefs to be dominated by particles generated by the proposed generator $\mathcal{G}(\mathbf{s}_t, a_t)$, rather than particles obtained from real-world observations.

We compare the results obtained when the tree depth is increased from 2 to 5, while keeping the number of particles the same.
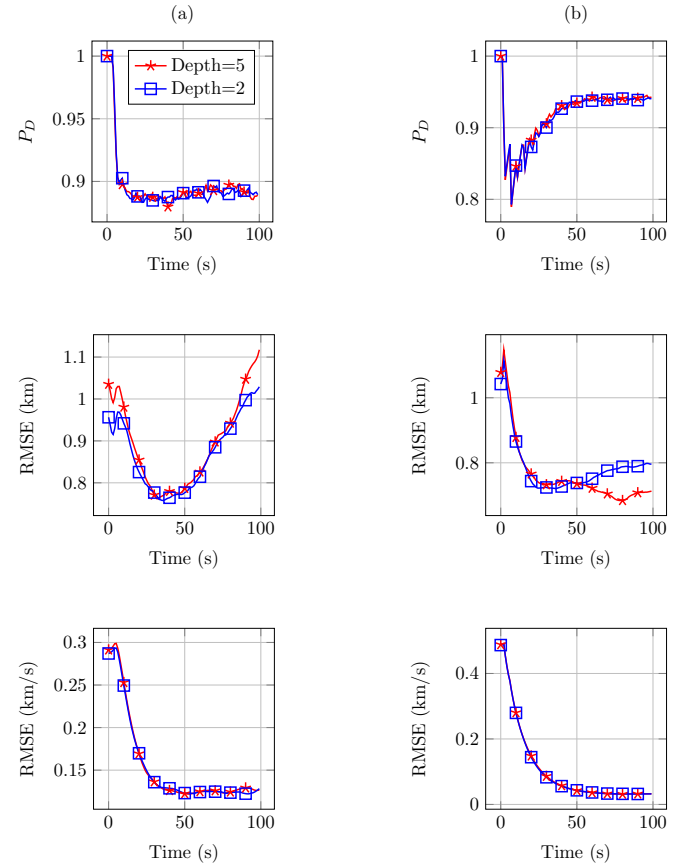

Fig. 10: Results obtained by increasing tree depth from 2 to 5. (a) Study case 1, (b) Study case 2.

Figure 10 demonstrates the effects of tree depth. Increasing the depth from 2 to 5 slightly improved Cartesian coordinate estimation in study case 2. In the other study case, a deeper tree caused a slight decline in position estimation accuracy,

while the P_D and velocity estimation remained unaffected in both study cases.

As an illustration, in the simulations, smaller tree depths are more suitable for the highly dynamic target (study case 1). In contrast, stable trajectories in study case 2 may have benefited from higher tree depths.

D. Actions taken by the POMCP

This section aims to demonstrate the actions taken by the POMCP and how they evolve over time. Given that the particle filter performed badly, this section will only compare the POMCP actions against the optimal actions. In essence, the POMCP has access to $L = 100$ actions, which is to select the potential angle bin θ_l where the target could be, and after that, the waveform matrix is computed given the chosen angle bin. This suggests that optimal actions depend only on the trajectory of the target. Figure 11 confirms this suggestion as the six examples show that the optimal action the radar should choose changes over time.
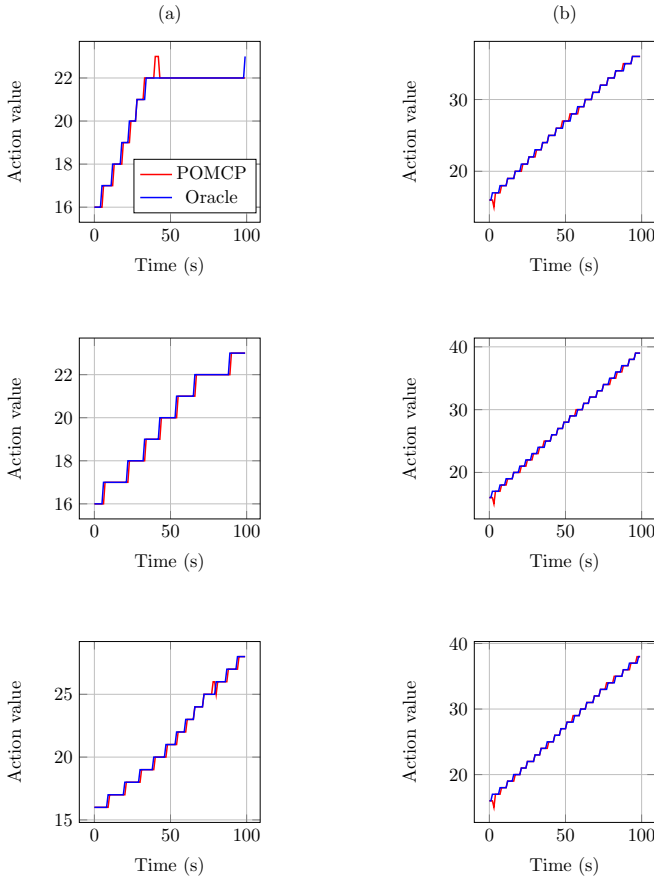

Fig. 11: Examples of actions chosen by the POMCP compared to the optimal actions. (a) Study case 1, (b) Study case 2.

E. Other hyperparameters

The algorithm involves numerous hyperparameters, some influencing the radar's number of virtual channels N, while others control the POMCP algorithm's behavior. Additional simulations were conducted by varying the P_{FA} and the clutter

disturbance distribution (using a Generalised Gaussian for the innovations). However, these variations yielded results consistent with our previous findings, with the POMCP successfully tracking and detecting the target. Given the similarity of these outcomes to the presented results, we opted not to include these additional simulations in the paper.

VI. Conclusion

This work presents an original POMCP framework for the joint detection and tracking of a moving target in disturbance-agnostic scenarios for Massive MIMO radar systems. Our results show that POMCP maintains reliable detection probabilities even under unknown disturbances, successfully tracking targets across varying velocities. This study validates the POMCP as a practical solution for real-world radar applications. Future work will focus on developing a comprehensive understanding of the impact of hyperparameters on algorithm performance. Additionally, efforts will be directed towards extending this framework to multi-target scenarios. This extension would involve adapting the POMDP model to handle multiple state spaces and exploring more efficient particle sampling strategies to manage the increased computational complexity while ensuring real-time performance.

References

[1] S. Haykin, "Cognitive Radar: a Way of the Future," *IEEE Signal Processing Magazine*, vol. 23, no. 1, pp. 30–40, 2006.

[2] S. Haykin, Y. Xue, and P. Setoodeh, "Cognitive Radar: Step Toward Bridging the Gap Between Neuroscience and Engineering," *Proceedings of the IEEE*, vol. 100, no. 11, pp. 3102–3130, 2012.

[3] S. P. Sira, Y. Li, A. Papandreou-Suppappola, D. Morrell, D. Cochran, and M. Rangaswamy, "Waveform-agile sensing for tracking," *IEEE Signal Processing Magazine*, vol. 26, no. 1, pp. 53–64, 2009.

[4] M. Hurtado, T. Zhao, and A. Nehorai, "Adaptive Polarized Waveform Design for Target Tracking Based on Sequential Bayesian Inference," *IEEE Transactions on Signal Processing*, vol. 56, no. 3, pp. 1120–1133, 2008.

[5] M. S. Greco, F. Gini, P. Stinco, and K. Bell, "Cognitive Radars: On the Road to Reality: Progress Thus Far and Possibilities for the Future," *IEEE Signal Processing Magazine*, vol. 35, no. 4, pp. 112–125, 2018.

[6] S. Z. Gurbuz, H. D. Griffiths, A. Charlish, M. Rangaswamy, M. S. Greco, and K. Bell, "An Overview of Cognitive Radar: Past, Present, and Future," *IEEE Aerospace and Electronic Systems Magazine*, vol. 34, no. 12, pp. 6–18, 2019.

[7] S. Fortunati, L. Sanguinetti, F. Gini, M. S. Greco, and B. Himed, "Massive MIMO Radar for Target Detection," *IEEE Transactions on Signal Processing*, vol. 68, pp. 859–871, 2020.

[8] A. M. Ahmed, A. A. Ahmad, S. Fortunati, A. Sezgin, M. S. Greco, and F. Gini, "A Reinforcement Learning Based Approach for Multitarget Detection in Massive MIMO Radar," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 57, no. 5, pp. 2622–2636, 2021.

[9] F. Lisi, S. Fortunati, M. S. Greco, and F. Gini, "Enhancement of a State-of-the-Art RL-Based Detection Algorithm for Massive MIMO Radars," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 58, p. 5925–5931, Dec. 2022.

[10] K. L. Bell, C. J. Baker, G. E. Smith, J. T. Johnson, and M. Rangaswamy, "Cognitive Radar Framework for Target Detection and Tracking," *IEEE Journal of Selected Topics in Signal Processing*, vol. 9, no. 8, pp. 1427–1439, 2015.

[11] P. Chavali and A. Nehorai, "Scheduling and Power Allocation in a Cognitive Radar Network for Multiple-Target Tracking," *IEEE Transactions on Signal Processing*, vol. 60, no. 2, pp. 715–729, 2012.

[12] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and Acting in Partially Observable Stochastic Domains," *Artif. Intell.*, vol. 101, p. 99–134, May 1998.

[13] M. J. Hausknecht and P. Stone, "Deep Recurrent Q-Learning for Partially Observable MDPs," *CoRR*, vol. abs/1507.06527, 2015.

[14] L. Kocsis and C. Szepesvári, "Bandit Based Monte-Carlo Planning," in *Machine Learning: ECML 2006* (J. Fürnkranz, T. Scheffer, and M. Spiliopoulou, eds.), (Berlin, Heidelberg), pp. 282–293, Springer Berlin Heidelberg, 2006.

[15] D. Silver and J. Veness, "Monte-Carlo Planning in Large POMDPs," in *Proceedings of the 23rd International Conference on Neural Information Processing Systems - Volume 2*, NIPS'10, (Red Hook, NY, USA), p. 2164–2172, Curran Associates Inc., 2010.

[16] Y. Huang, R. Guo, Y. Zhang, and Z. Chen, "Deep Reinforcement Learning Based Radar Parameter Adaptation for Multiple Target Tracking," *IEEE Transactions on Aerospace and Electronic Systems*, vol. PP, pp. 1–18, 01 2024.

[17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is All you Need," *CoRR*, vol. abs/1706.03762, 2017.

[18] M. Kozy, J. Yu, R. M. Buehrer, A. Martone, and K. Sherbondy, "Applying Deep-Q Networks to Target Tracking to Improve Cognitive Radar," in *2019 IEEE Radar Conference (RadarConf)*, pp. 1–6, 2019.

[19] Q. Wang, Y. Qiao, and L. Gao, "A Cognitive Radar Waveform Optimization Approach based on Deep Reinforcement Learning," in *2019 IEEE International Conference on Signal, Information and Data Processing (ICSIDP)*, pp. 1–6, 2019.

[20] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing Atari with Deep Reinforcement Learning," 2013. cite arxiv:1312.5602Comment: NIPS Deep Learning Workshop 2013.

[21] B. W. Rybicki and J. K. Nelson, "A Cognitive Tracking Radar using Continuous Space Monte Carlo Tree Search," in *2022 IEEE Radar Conference (RadarConf22)*, pp. 1–6, 2022.

[22] Z. Sunberg and M. J. Kochenderfer, "POMCPOW: an Online Algorithm for POMDPs with Continuous State, Action, and Observation Spaces," *CoRR*, vol. abs/1709.06196, 2017.

[23] B. W. Rybicki and J. K. Nelson, "Voronoi Progressive Widening for Cognitive Radar Tracking with Large Waveform Libraries," in *2024 IEEE Radar Conference (RadarConf24)*, pp. 1–6, 2024.

[24] M. H. Lim, C. J. Tomlin, and Z. N. Sunberg, "Voronoi Progressive Widening: Efficient Online Solvers for Continuous State, Action, and Observation POMDPs," in *2021 60th IEEE Conference on Decision and Control (CDC)*, pp. 4493–4500, 2021.

[25] P. Stoica, J. Li, and Y. Xie, "On Probing Signal Design for MIMO Radar," *IEEE Transactions on Signal Processing*, vol. 55, no. 8, pp. 4151–4161, 2007.

[26] A. M. Ahmed, L. Gharsalli, S. Fortunati, and A. Sezgin, "Reinforcement Learning for Cognitive Integrated Communication and Sensing Systems," in *2023 20th European Radar Conference (EuRAD)*, pp. 395–398, 2023.

[27] S. Fortunati, L. Sanguinetti, F. Gini, M. S. Greco, and B. Himed, "Erratum to "Massive MIMO radar for target detection"," *IEEE Transactions on Signal Processing*, vol. 69, pp. 3235–3235, 2021.

[28] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. The MIT Press, second ed., 2018.

[29] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time Analysis of the Multiarmed Bandit Problem," *Machine Learning*, vol. 47, pp. 235–256, May 2002.

[30] H. Zhang, J. Xie, J. Shi, Z. Zhang, and X. Fu, "Sensor Scheduling and Resource Allocation in Distributed MIMO Radar for Joint Target Tracking and Detection," *IEEE Access*, vol. 7, pp. 62387–62400, 2019.