arXiv:2306.11301v2 [cs.LG] 21 Oct 2023

# Adversarial Search and Tracking with Multiagent Reinforcement Learning in Sparsely Observable Environment

Zixuan Wu[1,†], Sean Ye[1,†], Manisha Natarajan[1], Letian Chen[1], Rohan Paleja[1], and Matthew C. Gombolay[1]

***Abstract*— We study a search and tracking (S&T) problem where a team of dynamic search agents must collaborate to track an adversarial, evasive agent. The heterogeneous search team may only have access to a limited number of past adversary trajectories within a large search space. This problem is challenging for both model-based searching and reinforcement learning (RL) methods since the adversary exhibits reactionary and deceptive evasive behaviors in a large space leading to sparse detections for the search agents. To address this challenge, we propose a novel Multi-Agent RL (MARL) framework that leverages the estimated adversary location from our learnable filtering model. We show that our MARL architecture can outperform all baselines and achieves a 46% increase in detection rate.**

## I. Introduction

Search and tracking (S&T) is a fundamental problem that requires autonomous agents to work together to find and track a specific object of interest, which is critical for a wide range of applications, from drug smuggling tracking to endangered wildlife monitoring. It is often difficult and cumbersome to design hand-crafted expert S&T policies as it is intractable to manually abstract patterns from complex environments with limited knowledge of adversarial target motions perfectly. Therefore, we propose a statistic-driven MARL framework with a Prior Motion Combined (PMC) filter model to perform informed exploration and maximize performance in a complex sparsely observable environment.

In this paper, we focus on the challenging adversarial S&T setting of reactive target and sparse detection, which means our heterogeneous search team can only make decisions based on intermittent information. Previous non-learning based methods have been developed to provide fast, robust and accurate tracking capabilities [1]–[5], which sometimes requires auxiliary parallel and hierarchical architectures, including probability map construction [6], detection-object association [7]–[9], target location filtering [7], [8], [10], and searching strategy optimization [6], [11]. The appropriate design, organization, and coordination of such subsystems can improve the S&T efficiency, but none of these is designed for **adversarial** S&T. In recent years, MARL has shown promising results in learning S&T policies by outperforming hand-tuned expert heuristics [11]. Several approaches have attempted to tackle exploration in sparse reward environments through reward shaping [12] or experience replay mechanisms [13], [14]. However, these methods still rely on expert-designed rewards, goals, or repeated environment replay from specific states. In contrast, we propose to address this challenge using opponent modeling, which involves reasoning about the opponent's strategy to enhance decision-making [15].

In this work, we present a novel approach that leverages estimated adversarial locations from the PMC filter to improve the efficiency of MARL for S&T in large adversarial environments with sparse detections. The PMC filtering model consists of a mixture-density network (MDN) that balances prior knowledge with a forward-motion model and is trained from past information we have about the target (e.g. wild animal behavior patterns [16] or drug smuggling routine [17] etc.) The key idea is to enhance the RL process by introducing a filtering model, which serves as a shortcut for inferring the evading agent's location.

**Contributions:** In summary, our contributions are three-fold:

1) We design a concise PMC filter structure that combines an *a priori* model with a linear forward motion model, demonstrating its effectiveness in localizing targets.
2) We propose a data-driven framework, utilizing the PMC filter in a MARL framework for a heterogeneous S&T team to track an intelligent evasive target in low observable settings (Figure 1).
3) We showcase the effectiveness of our approach in tracking adversarial targets compared to baseline filters, expert policies, and other MADDPG baselines. Our method achieves an increase of 24.7%, 46%, and 51% in localization accuracy, detection rate, and tracking performance, respectively.

## II. Related Work

### A. Object Search and Track

Target search and tracking (S&T) has emerged as a prominent research topic in recent years [1], [6], [9]–[11], [18]–[22]. Researchers have approached this problem by creating heuristic search strategies for agents such as spirals, lawnmowers [19], or interceptions [20]. However, these methods either assume knowledge of the target's exact location [20], rely on a wide field of view [1], or are designed for non-evading, non-adversarial targets [6], [21]. Other work that assumes dynamic targets utilizes an exploration-exploitation dynamics (EED) strategy [2] based upon a particle swarm

*This work is supported in part by the Office of Naval Research (ONR) under grant numbers N00014-19-1-2076, N00014-22-1-2834, and N00173-21-1-G009, the National Science Foundation under grant CNS-2219755, and MIT Lincoln Laboratory grant number 7000437192.

† Equal contribution

[1]All authors are associated with the Institute of Robotics and Intelligent Machines (IRIM), Georgia Institute of Technology, Atlanta, GA 30308, USA.

Correspondance Author: Zixuan Wu zwu380@gatech.edu

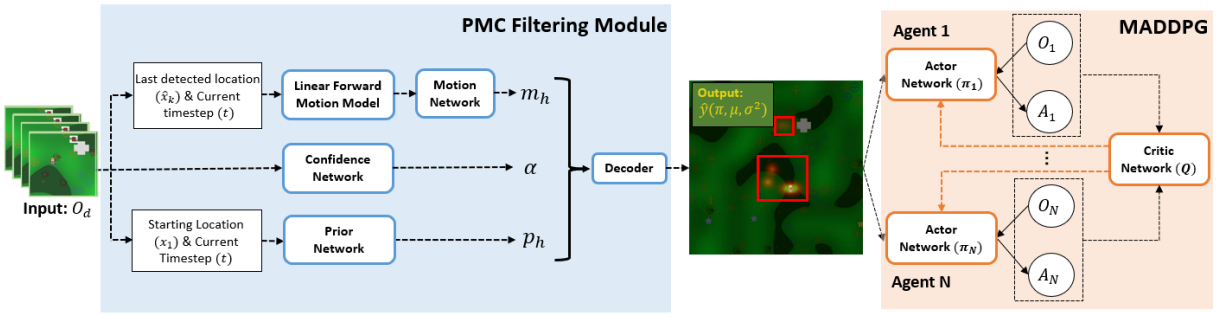



Fig. 1. Proposed Architecture (PMC-MADDPG): We augment the observations of each search agent with outputs from our Prior-Motion Combined (PMC) filtering model - gold regions (e.g. in the red boxes) indicate multiple output Gaussians on the map. The PMC is warmstarted offline and trained using the negative log likelihood with a set of pre-collected trajectories. We then utilize MADDPG to train the searching agents to track the opponent agent.

optimization algorithm [3] or information theoretic objective functions [23], [24]. In this paper, we demonstrate that our proposed learning-based approach surpasses both hand-crafted heuristics and swarm searching strategies.

### B. Opponent Modeling

Opponent modeling usually uses neural networks for target or opponent prediction. For instance, DRON is proposed to model the behavior of opponents from opponent observations [25]. Grover et al. use imitation learning and contrastive learning to predict the next actions of an opponent [26]. Yu et al. utilize a model-based approach with neural networks for predicting the next state of the opponents [27]. However, such approaches assume full observability of the opponent.

In this paper, we propose a neural network filtering module (PMC) that estimates the current opponent location from limited detection and knowledge generalized from its previous behavior, to help the downstream MARL track an evasive target. Although traditional non-learning-based filters [7], [8], [28], [29] have been widely used for agent localization, such classic filters can neither incorporate prior knowledge nor generalize well for under-determined observation conditioning in long term settings since they usually require the difference between consecutive predictions [7], [10], [30].

### C. Multi-Agent Reinforcement Learning (MARL)

MARL algorithms are designed to train multiple agents jointly to solve tasks in a shared environment. The simplest form is to train each agent independently with traditional RL algorithms (e.g. DQN [31], DDPG [32]). However, such an approach for training agent policies is not guaranteed to converge in a non-stationary multi-agent system.

Rather than training each agent independently, researchers have utilized centralized training decentralized execution (CTDE) algorithms where all agent observations are used to learn a centralized critic, but each agent uses its own observations during execution [33], [34]. Other solutions include utilizing learnable online communication channels such that agents can learn whom to communicate with, how to process messages, and when to communicate [35], [36]. We build upon MADDPG [33] in our approach to train S&T agents in an adversarial domain.

Utilizing MARL in adversarial environments is difficult as the policy of the opponent may evolve during training which leads to a distribution shift in the optimal policy for the search agents. AlphaStar [37] partially solves the issue by training a "league" of agents corresponding to different strategies such that each individual policy can focus on optimizing a certain aspect of the strategy space. Other approaches have used opponent modeling [25]–[27], [38] to predict the future location of adversarial agents as additional loss terms during training. However, these approaches typically rely on continuous knowledge of opponent observations as input, which is not applicable in our setting. Our learnable PMC filter only conditions on the previous sparse detections and improves tracking performance on a continuous action, large state space, and partially observable domain compared to previous discrete action, small state space, and fully observable domains commonly used for MARL.

## III. Environment

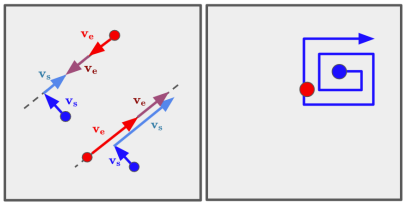
### A. Partially Observable Markov Game

A Markov Game is the multi-agent version of a Markov Decision-Process (MDP). In this work, we model adversarial search and track as a Partially Observable Markov Game (POMG) [39]. A POMG for N-agents can be defined by a set of states $\mathcal{S}$, a set of private observations for each agent $\{\mathcal{O}_1, \mathcal{O}_2, \ldots, \mathcal{O}_N\}$, a set of actions for each agent $\{\mathcal{A}_1, \mathcal{A}_2, \ldots \mathcal{A}_N\}$, a transition function $\mathcal{T} : \mathcal{S} \times \mathcal{A}_1 \times \ldots \times \mathcal{A}_N \mapsto \mathcal{S}$. At each timestep $t$, every agent indexed $i, i \in \{1, \ldots, N\}$ receives an observation $O_i^t$, chooses an action $A_i^t \in \mathcal{A}_i$, and obtains a reward $r_i^t : \mathcal{S} \times \mathcal{A}_i \mapsto \mathbb{R}$. The initial world state is drawn from a prior distribution $\rho$. In our environment (see §III-B), the RL policy has to explore a large search space with nearly no detection at the start, and the observation distribution is approximately static. However, the distribution will change when the policy becomes better and the detections increase. We expect our designed filter can help RL for an easier transition.

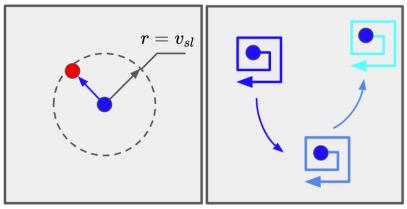
### B. Fugitive Escape Scenario

In our evaluation of the proposed algorithm for adversarial S&T, we use a large-scale ($2428 \times 2428$, representing 51 x 51 km) pursuit-evasion domain named Fugitive Escape scenario [40][1]. An intelligent evader with max speed 15 per timestep is being tracked by a team of search agents in a

[1]https://github.com/CORE-Robotics-Lab/Opponent-Modeling




(a) Intercept (b) Spiral




(c) Pointwise (d) Random Spiral

Fig. 2. Subpolicies used in the search agent heuristics.

partially observable environment. The game-ending goal of the evader is to reach any hideout on the map, of which the pursuit team only knows a partial set. The terrain has varying visibility levels, with dense forest regions hindering the agents' detection abilities. Using $A^*$ search, the evader covertly makes their way to a hideout while traversing the shortest distance and taking advantage of dense cover. Additionally, the evader employs dynamic planning by using evasive behaviors to escape from the search team's contact.

The heterogeneous pursuit team consists of static (cameras) and dynamic agents (search parties and helicopters with max speed 20 and 127 respectively) that collaborate to search and track the evader. Both the evading agent and the pursuit agents can only detect each other within a certain range, determined by their speeds and the terrain.

### C. Heuristic Policy

We create a heuristic policy for tracking the evader inspired by previous object searching works [19], [20]. The search agents can choose to either 1) go to the last known location of the evader, 2) intercept the evader, or 3) search in the vicinity of the evader's last known location.

In the Fugitive Escape domain, we have two types of dynamic agents— search parties and helicopters, with speed limits of $v_{sl}$ and $v_{hl}$ respectively. The detection history of the evader is shared across all search agents and is denoted as $\{d_i\}_{i=1}^{N_d}$, where $d_i$ is the detected location of the evader and $N_d$ is the number of detections from the start of the episode. We denote the current detected location of the evader as $x_p$ and that of any search agent as $x_s$.

If the evader is detected nearby and within reaching distance, (i.e. $\|x_p - x_s\|_2 < v_{sl}$), the search agent will directly go to the detected location with the pointwise policy (Figure 2c). Otherwise, the search agent will intercept by going to the interception point perpendicular to the opponent agent velocity (Figure 2a) and then move along the vector towards the opponent's estimated location. If the search agent does not detect the evader, it will spiral around the estimated evader location for a set time (Figure 2b). If the agent still fails to find the evader, then the policy is reset and the search agent starts a spiral search at a random location (Figure 2d).

## IV. Method

We propose a novel approach named Prior Motion Combined MADDPG (PMC-MADDPG) to improve the efficiency of solving S&T problems in sparse detection environments (Figure 1). PMC-MADDPG is a framework that embeds a filtering model (PMC), which estimates the current location of the adversary, into MADDPG. We show that the filtering model aids in improving MARL performance even with sparse information at the start of training (§V). We take inspiration from classical filters, where most classical methods merge different information sources (e.g. prediction and update, multiple motion models, etc.) with weights (e.g. Kalman gain) for some optimal loss (minimum mean-square error). Following this pattern, we build our filter as two different information branches (a prior network and a linear motion model) and a learnable weight $\alpha$.

### A. PMC Filter System Design

Our proposed Prior Motion Combined (PMC) filtering model (Figure 1) consists of three parts 1) a learned prior network to inform the search agents of the adversarial agent preferences, 2) a linear motion model predictor to provide estimates of the adversarial motion and 3) a confidence network that learns to weigh these components relative to the input observation.

*1) Prior Network:* The adversary may exhibit distinct patterns or preferences while meeting its objectives such as evading surveillance agents, optimizing travel costs or heading to different hideouts. We utilize a learning based approach to encode data from past trajectories of the adversary into an embedding space. Fully connected (FC) network is used as an encoder that takes as input the starting location of the adversary and the current timestep from the observation:

$$\boldsymbol{p_h} = Encoder(\boldsymbol{x_1}, \boldsymbol{t}) \tag{1}$$

where $\boldsymbol{p_h}$ is the embedding from the encoder, $\boldsymbol{x_1}$ is the starting location of the adversary and $t$ is the current timestep.

*2) Motion Model:* We use a linear motion model where the input is the timestep and the last two detected locations. We output the estimated current location based on the constant velocity assumption. The state of the adversarial agent at timestep $k$ can be estimated as:

$$\boldsymbol{x}_k = \boldsymbol{x}_{\hat{k}} + (k - \hat{k})\boldsymbol{u}_{\hat{k}} \tag{2}$$

where $\boldsymbol{x}_k$ and $\boldsymbol{u}_k$ are the state (location) and control input (velocity); $\hat{k}$ is the latest timestep where detection occurred. This can be shown to satisfy the intermittent Kalman filter estimation [30]. We pass the output of the motion model $\boldsymbol{x_k}$ through several fully connected layers to produce an embedding $\boldsymbol{m_h}$ for the motion prediction to be used for the decoder described in the following subsection §IV-A.3.

*3) Confidence Network:* We use an FC *confidence net* to generate a scalar weight $\alpha$ from the observation to balance the prior and motion outputs:

$$\boldsymbol{G}_e = \alpha \boldsymbol{p}_h + (1-\alpha)\boldsymbol{m}_h \tag{3}$$

where $\boldsymbol{p}_h$, $\boldsymbol{m}_h$ and $\boldsymbol{G}_e$ are prior, motion and Gaussian embeddings respectively and $\alpha$ is the learnable weight.

Since the adversary's path preferences are encoded in the prior network, the decoder can produce multiple hypotheses (mixture of Gaussians) for state estimation:

$$\boldsymbol{\lambda}_i, \boldsymbol{\mu}_i, \boldsymbol{\sigma}_i = Decoder(\boldsymbol{G}_e), i = 1, 2, 3, ..., N_g \tag{4}$$

where $N_g = 8$ is the number of components in the Gaussian mixture and $\boldsymbol{\lambda}_i$, $\boldsymbol{\mu}_i$ and $\boldsymbol{\sigma}_i$ are the mixture weight, mean and variance of the mixture component $i$. We utilize the negative log-likelihood loss (Equation 5) for a mixture of Gaussians to train the entire network including the prior network, layers encoding the motion model, and decoder:

$$\mathcal{L} = -\log(\sum_{i=1}^{N_g} \boldsymbol{\lambda_i} \cdot p(\boldsymbol{y}|\boldsymbol{\mu_i}, \boldsymbol{\sigma_i}^2)) \tag{5}$$

where $\boldsymbol{y}$ is the true adversary location normalized to $[0, 1]$.

### *B. PMC Filter Informed MARL*

In this section, we discuss how the PMC Filter described in Section IV-A is utilized in the MARL formulation to improve search and tracking. We hypothesize that our RL based method (PMC-MADDPG) can perform better than hand crafted heuristics [19], [20] because the policy is capable of adapting to changing environment state distributions.

In traditional MADDPG, the value function faces the challenge of inferring the evasive agent's location from observations and rewards in a noisy and sparsely observable environment. This inference becomes particularly difficult due to the limited exploration of the state space. However, our PMC filter, trained directly with a supervised loss using historical trajectories, offers a solution. By augmenting the input to the actor-critic with an estimate of the evading agent's location, the PMC model helps alleviate the burden of this challenging inference. As a result, we propose that the PMC model effectively informs MARL training and enhances both sample efficiency and exploration capabilities.

*1) Multi-agent deep deterministic policy gradient (MADDPG):* MADDPG is a multi-agent extension of the basic Deep Deterministic Policy Gradient (DDPG) reinforcement learning algorithm and follows the off-policy centralized training and decentralized execution paradigm [33]. Each agent has its own actor-critic structure but the critic takes the combined state-action pairs of all agents compared with the input of the actor which is each its own experience tuple.

We use MADDPG as our framework to train the search agent policies. For each search agent, the base observation $o_b$ includes its own current location, the current timestep, the current locations of all other searching agents, and the location of known hideouts. The augmented observation $o_a$ is the extra information appended to each agent's observation. The PMC-MADDPG uses the estimated opponent location from the filter relative to the search agent as $o_a$ (Equation (6)):

$$\mu_j^{loc} = \mu_j - x_s, j = 1, 2, 3, ..., N_g \tag{6}$$

Each agent $i$ receives its own observation denoted as $o_i = [o_b \| o_a]$. Based on this observation, each agent independently determines its action using its own FC deterministic policy:

$$a_i = \pi_i(o_i), i = 1, 2, ..., N_a \tag{7}$$

where $N_a$ is the number of search agents. The critic is then updated to minimize the TD error ($L(\theta_i)$) between two consecutive timesteps:

$$L(\theta_i) = \mathbb{E}_{O,a,r,O'}[(Q_i^{\pi}(O, a_1, ..., a_{N_a}) - y)^2] \tag{8}$$

$$y = r_i + \gamma Q_i^{\pi'}(O', a'_1, ..., a'_{N_a})|_{a'_i = \pi'_i(o_i)} \tag{9}$$

Then the policy is updated with the gradient:

$$\nabla_{\xi_i} J(\xi_i) = \mathbb{E}_{O,a\sim\mathcal{D}}[\nabla_{\xi_i}\pi_i(a_i|o_i) \nabla_{a_i} Q_i^{\pi}(O, a_1, ..., a_N)|_{a_i=\pi_i(o_i)}] \tag{10}$$

where $\gamma$ is discount factor. $\theta_i, \xi_i, i = 1, 2, ..., N_a$ are the parameters of the critic and actor neural networks $Q_i, \pi_i$ respectively. $O$ is the concatenation of all agents' observation. We assume the action $a_i$ to be the search agent velocity. Samples are randomly drawn from the replay buffer $\mathcal{D}$ to train the actor-critic and all the gradients are cut off at the observation, which means we only update the network parameters using equations (9) and (10) while the PMC is updated only with (5). The superscript prime means the quantity is of the next timestep and the subscript indicates the agent index. We train the algorithm with each agent's detection reward and a distance shaping term as [33].

*2) PMC Filter in MADDPG:* The evading agent is rarely detected at the start of the MARL because the search agent policies are not trained well. Therefore we hypothesize that our PMC filter can compensate for the information loss by providing its position estimation to foster training. Specifically, we concatenate the Gaussian parameters $O_a = \{\lambda_i, \mu_i, \sigma_i\}_{i=1}^{N_g}$ from the filter to the end of the base observation $O_b$ as described in §IV-B.1 and compare the results with other augmented observations.

We evaluate the benefit of augmenting MADDPG with a filter module by comparing the performance on two datasets – random and heuristic depending on how the search agents interact with the evader when pretraining the filter. In the **random** dataset, the search agents use random policies such that there are nearly no detections and the evader does not need to actively escape. In the **heuristic** dataset, the search agents use heuristics such that the evader has interactions with the searching team. We show that our approach outperforms all baselines in both datasets in §V.

## V. Experiments and Results

This section shows our proposed PMC filter model can successfully estimate the evader location and aid in downstream MADDPG training. We divide our comparisons into

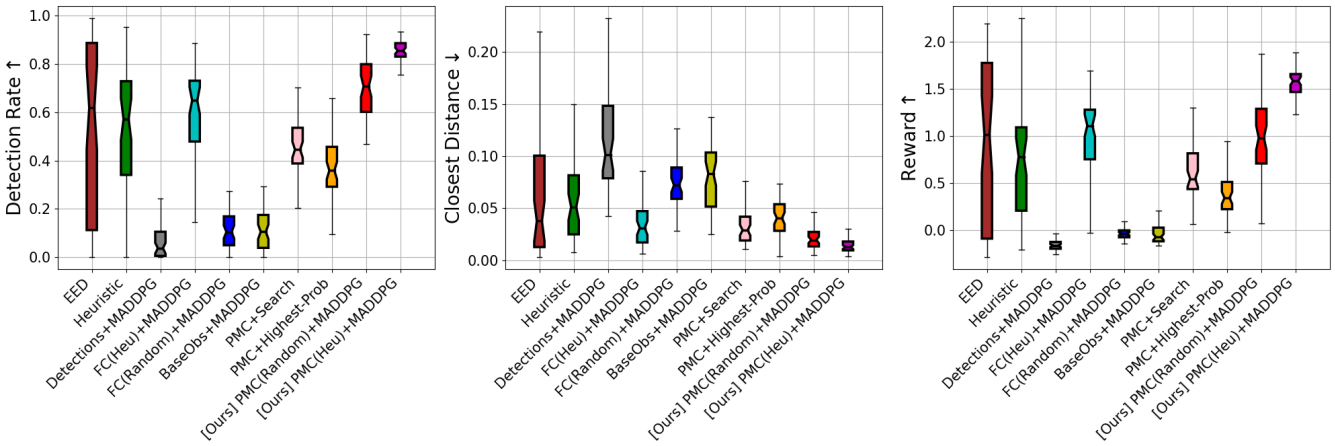



Fig. 3. MARL + Filtering evaluation results: we show our PMC filtering model + MADDPG outperforms all other baselines. Higher detection rate and reward, and lower closest distance means better performance.

three groups: 1) PMC filter vs. RNN-based baseline filters, 2) MARL-based method vs. heuristics, and 3) validation of the PMC filter's positive contribution to MARL. We report the MARL performance averaged over 50 trajectories and show all experiment results in Table III, Figures 3-4 where closest distance indicates the distance between the evader and the closest search agent (↓ means lower is better). We denote the search agent policies used to warm-start the filter as either a heuristic policy (Heu) or random policy (Random).

### A. PMC Filter Validation

We first assess the accuracy of our PMC filter by comparing it to several baseline filters on prior datasets independent of MARL. Our PMC filter outperforms four RNN-based filters and one FC configuration (Table I-II) on various metrics. RT refers to the forward running time for one batch. We evaluate the log-likelihood (LL) of the adversary location given the predicted Gaussian distribution. ADE is the $l_2$ distance between the mean of the predicted distribution and the adversary location. Confidence Threshold Percentage (CTP) is the ratio of timesteps where the model predicts a probability of the adversary location greater than $0.5$.

TABLE I

RANDOM SEARCH POLICY EVALUATION METRICS (MEAN)

| **Filt.** | RT↓ | LL↑ | ADE↓ | CTP↑ |
|---|---|---|---|---|
| LSTM | 8.93e-4 | 1.74 | 1.38e-1 | 4.27e-1 |
| GNN | 7.14e-3 | 2.93 | 7.91e-2 | 5.12e-1 |
| VRNN | 8.85e-3 | 4.70 | 4.59e-2 | 9.01e-1 |
| VRNN-GNN | 1.51e-2 | 3.61 | 6.13e-2 | 7.14e-1 |
| FC | **5.02e-4** | 5.73 | 5.74e-2 | 9.63e-1 |
| PMC [ours] | 9.17e-4 | **7.25** | **4.19e-2** | **9.71e-1** |

TABLE II

HEURISTIC SEARCH POLICY EVALUATION METRICS (MEAN)

| **Filt.** | RT↓ | LL↑ | ADE↓ | CTP↑ |
|---|---|---|---|---|
| LSTM | 8.88e-4 | 1.74 | 1.28e-1 | 4.43e-1 |
| GNN | 7.56e-3 | 4.74 | 5.23e-2 | 8.11e-1 |
| VRNN | 9.11e-3 | 5.13 | 5.46e-2 | 8.67e-1 |
| VRNN-GNN | 1.42e-2 | 5.33 | 7.69e-2 | 8.07e-1 |
| FC | **4.90e-4** | 6.15 | 4.16e-2 | 8.92e-1 |
| PMC [ours] | 9.24e-4 | **7.57** | **3.41e-2** | **9.66e-1** |

We evaluate all models on two datasets: one with random search agent policies (Table I) and one with heuristic search agent policies (Table II). All filter configurations follow an encoder-decoder structure where the encoder is indicated in Table I-II and the decoder is an FC layer that outputs the Gaussian mixture parameters. Both PMC and FC models use the same input (the last two detections with velocities, current timestep, and starting location of the evader). The input for all RNN-based filtering approaches additionally includes the observation histories of all agents. Results clearly show that only the FC model is comparable with our PMC model but is still worse by 19.84% and 30.26% on LL and ADE respectively, averaged across both datasets. Next, we show the results by incorporating the FC and PMC models with our MARL framework for adversarial S&T.

### B. Advantages of MARL over Heuristic Policies

We use two non-learning-based approaches as baselines—1) Heuristic, which combines commonly used search strategies like spirals, pointwise moving, and interceptions (see §III-C), and 2) Exploration Exploitation Dynamics (EED) which is a specialized dynamic target search and tracking [2] method. Both approaches have comparable results achieving greater than 50% detection rate. Our results show the MADDPG-based approaches outperform the non-learning-based heuristics by at least 15% and 36% in terms of detection rate and closest distance (Table III). We show that the learning-based MADDPG approaches are able to adapt better to the distribution of the adversarial environment compared to hand-crafted heuristics.

As an ablation study, we construct two hand-crafted search agent policies that use the PMC filter but *do not learn*. PMC(Heu) + Highest-Prob drives the search agents to the highest probability Gaussian while PMC(Heu) + Search splits the searching agents amongst multiple Gaussians. Our results show that the MADDPG policy can evolve as the distribution shifts (e.g., more evasive behavior when detected) compared to the deterministic heuristics even though they have same access to prior knowledge within the PMC filter.

### C. Model Based Filtering Benefits

We compare the results of utilizing a PMC model and those without a filter model. The BaseObs+MADDPG uses the null vector $\emptyset$ while Detections+MADDPG uses the last

TABLE III
COMBINATION OF DIFFERENT FILTERS AND POLICIES
(MEAN ± STANDARD DEVIATION)

| Filter & Policy | Detect. Rate | Closest Dist. | Reward |
|---|---|---|---|
| **Non-Learning Approaches** | | | |
| Heuristic | 0.52±0.28 | 0.06±0.05 | 0.82±0.73 |
| EED | 0.51±0.38 | 0.07±0.08 | 0.83±0.90 |
| PMC(Heu)+Highest-Prob | 0.38±0.16 | 0.04±0.02 | 0.44±0.40 |
| PMC(Heu)+Search | 0.47±0.13 | 0.03±0.01 | 0.67±0.38 |
| **Learning Approaches** | | | |
| BaseObs+MADDPG | 0.11±0.09 | 0.08±0.03 | -0.04±0.09 |
| Detections+MADDPG | 0.06±0.07 | 0.12±0.06 | -0.16±0.05 |
| FC(Random)+MADDPG | 0.11±0.08 | 0.08±0.03 | -0.04±0.05 |
| FC(Heu)+MADDPG | 0.60±0.19 | 0.04±0.03 | 1.01±0.42 |
| **Our Approches** | | | |
| PMC(Random)+MADDPG | 0.69±0.15 | 0.02±0.01 | 1.00±0.38 |
| PMC(Heu)+MADDPG | **0.84**±0.12 | **0.02**±0.04 | **1.53**±0.27 |

two detections (with velocities) $o_d = \{x_i, v_i\}_{i=1}^2$ as the augmented observation $o_a$. We show that our PMC models outperform these baselines, performing on average 88% better on detection rate and achieving 41 times as much reward. While the input to the agents is the same, the filtering models provide the location estimates from an extra network that the baselines do not have. We confirm that providing tabula rasa agents with an opponent model from detection history enables them to learn within our large, partially observable domain.

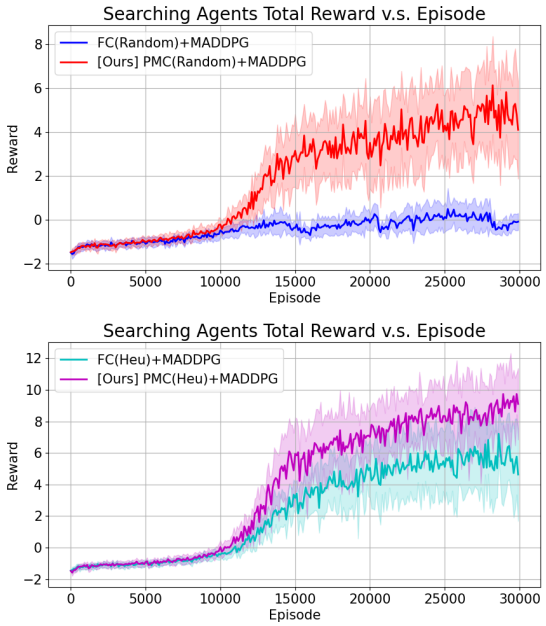



Fig. 4. We sum all searching agents rewards and plot the total reward curve versus the training episodes. The PMC based MADDPG gains reward faster and higher. The filters trained on heuristic dataset perform better than those trained on random dataset.

### D. Comparing PMC with FC Filters within MADDPG

In this section, we analyze how the PMC filtering model performs against the FC when used within MADDPG. We use FC and PMC augmented MADDPG, which are the best filters from Section V-A. We find that the FC+MADDPG models underperform our PMC+MADDPG models by 53.59% on detection rate. Thus, the improvement in the MADDPG performance is due to the PMC filter structure since both PMC and FC filters have the same inputs. Reward curves in Figure 4 show that the PMC+MADDPG achieves better sample efficiency and higher reward compared with FC+MADDPG. Further, the PMC filter trained on the heuristic dataset (PMC(Heu)) can achieve higher performance since the filtering module also encodes the interactions between search and evader agents.

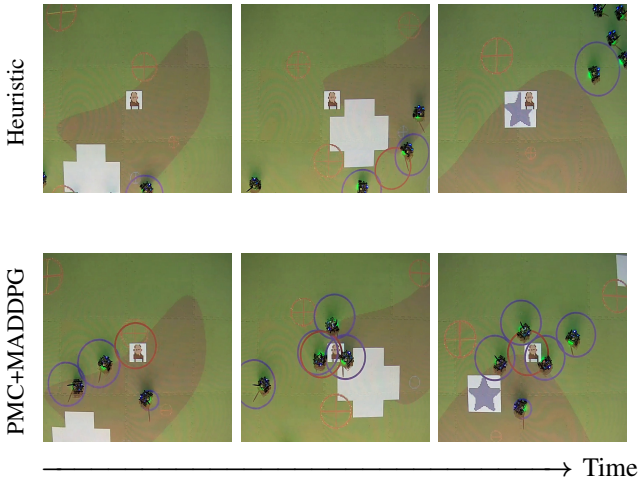



Fig. 5. Demonstration of Heuristic vs PMC+MADDPG(Heu) on real robots. The star and large white region represent the hideout and obstacle. Darker regions of the map represent dense forests where visibility is low.

### E. Real Robot Demonstration

We also conducted tests using the Robotarium [41] testbed to evaluate the performance of our PMC+MADDPG method in real-world scenarios with dynamic and collision constraints. The results, as depicted in Figure 5, highlight the comparison between the heuristic policy and our method. In these experiments, the ground robots represent the ground search parties, with blue circles indicating their detection range. Due to the unavailability of drones, we projected the helicopter (represented by a red circle) onto the test bed. The findings clearly demonstrate that the search agents following the heuristic policy fail to capture the evader by erroneously navigating to the wrong side of obstacles. Conversely, our PMC model, which proposes multiple potential evader positions, effectively assists the MARL policy in continuous tracking until the evader reaches their hideout.

## VI. DISCUSSION AND CONCLUSION

We propose a novel approach for improving the efficiency and effectivenes of S&T systems in sparse detection environments by leveraging a joint framework that utilizes a filtering module (PMC) within MADDPG. While the proposed approach demonstrated promising results, there exist a few limitations that can be addressed in future work. First, our current approach is aimed at tracking a single adversary, and we aim to extend our approach to handle multiple adversaries. Second, our current approach requires previous interactions to warmstart the filtering module, which

we intend to remove. We will also further try to apply the approach in drug tracing or disasters rescuing tasks.

## REFERENCES


[1] A. Banerjee, R. Ghods, and J. Schneider, "Multi-agent active search using detection and location uncertainty," *arXiv preprint arXiv:2203.04524*, 2022.

[2] H. L. Kwa, J. L. Kit, and R. Bouffanais, "Optimal swarm strategy for dynamic target search and tracking," in *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, 2020, pp. 672–680.

[3] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95-international conference on neural networks*, vol. 4. IEEE, 1995, pp. 1942–1948.

[4] R. C. Eberhart and Y. Shi, "Tracking and optimizing dynamic systems with particle swarms," in *Proceedings of the 2001 congress on evolutionary computation (IEEE Cat. No. 01TH8546)*, vol. 1. IEEE, 2001, pp. 94–100.

[5] Q. Liu, W. Wei, H. Yuan, Z.-H. Zhan, and Y. Li, "Topology selection for particle swarm optimization," *Information Sciences*, vol. 363, pp. 154–173, 2016.

[6] M. Meghjani, S. Manjanna, and G. Dudek, "Multi-target search strategies," in *2016 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, 2016, pp. 328–333.

[7] S. S. Blackman, "Multiple hypothesis tracking for multiple target tracking," *IEEE Aerospace and Electronic Systems Magazine*, vol. 19, no. 1, pp. 5–18, 2004.

[8] Y. Bar-Shalom and X.-R. Li, *Multitarget-multisensor tracking: principles and techniques*. YBs Storrs, CT, 1995, vol. 19.

[9] P. Rosello and M. J. Kochenderfer, "Multi-agent reinforcement learning for multi-object tracking," in *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, 2018, pp. 1397–1404.

[10] P. Dames, P. Tokekar, and V. Kumar, "Detecting, localizing, and tracking an unknown number of moving targets using a team of mobile robots," *The International Journal of Robotics Research*, vol. 36, no. 13-14, pp. 1540–1553, 2017.

[11] G. Wang, F. Wei, Y. Jiang, M. Zhao, K. Wang, and H. Qi, "A multi-auv maritime target search method for moving and invisible objects based on multi-agent deep reinforcement learning," *Sensors*, vol. 22, no. 21, p. 8562, 2022.

[12] J. Xie, Z. Shao, Y. Li, Y. Guan, and J. Tan, "Deep reinforcement learning with optimized reward functions for robotic trajectory planning," *IEEE Access*, vol. 7, pp. 105 669–105 679, 2019.

[13] Y. Du, G. Warnell, A. Gebremedhin, P. Stone, and M. E. Taylor, "Lucid dreaming for experience replay: refreshing past states with the current policy," *Neural Computing and Applications*, vol. 34, no. 3, pp. 1687–1712, 2022.

[14] Y. Zhou, Z. Liu, H. Shi, S. Li, N. Ning, F. Liu, and X. Gao, "Cooperative multi-agent target searching: a deep reinforcement learning approach based on parallel hindsight experience replay," *Complex & Intelligent Systems*, pp. 1–12, 2023.

[15] F. B. Von Der Osten, M. Kirley, and T. Miller, "The minds of many: Opponent modeling in a stochastic game." in *IJCAI*, 2017, pp. 3845–3851.

[16] S. M. Reader, J. R. Kendal, and K. N. Laland, "Social learning of foraging sites and escape routes in wild trinidadian guppies," *Animal Behaviour*, vol. 66, no. 4, pp. 729–739, 2003.

[17] M. Medel, Y. Lu, and E. Chow, "Mexico's drug networks: Modeling the smuggling routes towards the united states," *Applied geography*, vol. 60, pp. 240–247, 2015.

[18] Y. Zhou, A. Chen, X. He, and X. Bian, "Multi-target coordinated search algorithm for swarm robotics considering practical constraints," *Frontiers in Neurorobotics*, vol. 15, p. 753052, 2021.

[19] K. E. C. Booth, C. Piacentini, S. Bernardini, and J. C. Beck, "Target search on road networks with range-constrained uavs and ground-based mobile recharging vehicles," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6702–6709, 2020.

[20] A. I. A. Isaza, J. Lu, V. Bulitko, and R. Greiner, "A cover-based approach to multi-agent moving target pursuit," in *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, vol. 4, no. 1, 2008, pp. 54–59.

[21] G. Wang, F. Wei, Y. Jiang, M. Zhao, K. Wang, and H. Qi, "A multi-auv maritime target search method for moving and invisible objects based on multi-agent deep reinforcement learning," *Sensors*, vol. 22, no. 21, p. 8562, 2022.

[22] A. Afzalov, A. Lotfi, B. Inden, and M. E. Aydin, "A strategy-based algorithm for moving targets in an environment with multiple agents," *SN Computer Science*, vol. 3, no. 6, p. 435, 2022.

[23] P. Yang, Y. Liu, S. Koga, A. Asgharivaskasi, and N. Atanasov, "Learning continuous control policies for information-theoretic active perception," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 2098–2104.

[24] P. Yang, S. Koga, A. Asgharivaskasi, and N. Atanasov, "Policy learning for active target tracking over continuous $se(3)$ trajectories," in *Learning for Dynamics and Control Conference*. PMLR, 2023, pp. 64–75.

[25] H. He, J. Boyd-Graber, K. Kwok, and H. Daumé III, "Opponent modeling in deep reinforcement learning," in *International conference on machine learning*. PMLR, 2016, pp. 1804–1813.

[26] A. Grover, M. Al-Shedivat, J. Gupta, Y. Burda, and H. Edwards, "Learning policy representations in multiagent systems," in *International conference on machine learning*. PMLR, 2018, pp. 1802–1811.

[27] X. Yu, J. Jiang, W. Zhang, H. Jiang, and Z. Lu, "Model-based opponent modeling," *arXiv preprint arXiv:2108.01843*, 2021.

[28] T. Kurien, "Issues in the design of practical multitarget tracking algorithms," *Multitarget-multisensor tracking: advanced applications*, pp. 43–84, 1990.

[29] P. M. Dames, "Distributed multi-target search and tracking using the phd filter," *Autonomous robots*, vol. 44, no. 3-4, pp. 673–689, 2020.

[30] B. Sinopoli, L. Schenato, M. Franceschetti, K. Poolla, M. Jordan, and S. Sastry, "Kalman filtering with intermittent observations," *IEEE Transactions on Automatic Control*, vol. 49, no. 9, pp. 1453–1464, 2004.

[31] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[32] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[33] R. Lowe, Y. I. Wu, A. Tamar, J. Harb, O. Pieter Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," *Advances in neural information processing systems*, vol. 30, 2017.

[34] T. Rashid, M. Samvelyan, C. S. De Witt, G. Farquhar, J. Foerster, and S. Whiteson, "Monotonic value function factorisation for deep multi-agent reinforcement learning," *The Journal of Machine Learning Research*, vol. 21, no. 1, pp. 7234–7284, 2020.

[35] Y. Niu, R. Paleja, and M. Gombolay, "Multi-agent graph-attention communication and teaming," in *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*, ser. AAMAS '21. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, 2021, p. 964–973.

[36] E. Seraj, Z. Wang, R. Paleja, D. Martin, M. Sklar, A. Patel, and M. Gombolay, "Learning efficient diverse communication for cooperative heterogeneous teaming," in *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, ser. AAMAS '22. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, 2022, p. 1173–1182.

[37] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev, *et al.*, "Grandmaster level in starcraft ii using multi-agent reinforcement learning," *Nature*, vol. 575, no. 7782, pp. 350–354, 2019.

[38] S. Wu, T. Qiu, Z. Pu, and J. Yi, "Multi-agent collaborative learning with relational graph reasoning in adversarial environments," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 5596–5602.

[39] E. A. Hansen, D. S. Bernstein, and S. Zilberstein, "Dynamic programming for partially observable stochastic games," in *AAAI*, vol. 4, 2004, pp. 709–715.

[40] S. Ye, M. Natarajan, Z. Wu, R. Paleja, L. Chen, and M. C. Gombolay, "Learning models of adversarial agent behavior under partial observability," *Proceedings of the International Conference on Intelligent Robots and Systems*, 2023, to appear.

[41] S. Wilson, P. Glotfelter, L. Wang, S. Mayya, G. Notomista, M. Mote, and M. Egerstedt, "The robotarium: Globally impactful opportunities, challenges, and lessons learned in remote-access, distributed control of

multirobot systems," *IEEE Control Systems Magazine*, vol. 40, no. 1, pp. 26–44, 2020.