2024 American Control Conference (ACC)
July 8-12, 2024. Toronto, Canada

Deep Reinforcement Learning Based Distributed Active Joint Localization and Target Tracking

Dongming Wang[1], Shaoshu Su[2], Wei Ren[1], Ce Hao[3]

Abstract— **In this paper, we investigate the problem of active joint localization and target tracking of mobile robots with onboard sensors. Our primary objective is concurrent target tracking while precisely localizing the robots through coordinated motion. A key constraint is the distributed setting, where each robot's observations are limited to its immediate vicinity, and communication is restricted to neighboring robots.**

To address this, we propose a novel reinforcement learning-based approach for active motion planning, grounded in a distributed estimation framework called Joint Localization and Target Tracking (JLATT). The policy is trained to optimize robot coordination and trajectories for enhanced self-localization, target tracking, and collision avoidance. Empirical analysis demonstrates our algorithm's effectiveness compared to benchmarks, both in collision avoidance and reducing estimation covariance, affirming its robustness for complex robotic systems.

I. Introduction

Autonomous multi-robot systems, crucial in applications like search and rescue, region monitoring, and area surveillance, grapple with challenges in accurate positioning without GPS, motion capture, and prior maps. Joint Localization and Target Tracking (JLATT) involve simultaneous estimation of robot and target states. Existing methods, whether centralized or distributed, often assume sensors with static or random motion, lacking active control [1]–[5]. Algorithms in control often assume knowledge of both robot and target states, relying on centralized or multi-hop information transmission, impractical in real-world scenarios [6]–[11]. To address the Active JLATT (AJLATT) problem, strategies propose minimizing uncertainty in the target state while considering limited sensing capabilities. However, reliance on centralized estimation frameworks and strict communication topology requirements limit practicality [12], [13]. Even in distributed approaches, multi-hop information transmission remains a requirement [14].

Reinforcement learning (RL) has emerged as a promising approach for motion planning, exhibiting enhanced generalization through interactive learning [15]. This has garnered attention in multi-agent motion planning (MAMP). RL-based MAMP considers factors like local observability and environmental uncertainty, often extending to partially observable Markov decision processes (POMDPs) or decentralized POMDPs (Dec-POMDPs). RL-based MAMP research can be categorized into centralized (C-MAMP) and decentralized (D-MAMP) approaches. C-MAMP employs a centralized value network to learn a joint planning policy [15]. D-MAMP involves independent planning or centralized training with decentralized execution (CTDE), where robots access global information during training [16], [17]. Algorithms like CADRL (Collision Avoidance with Deep RL algorithm) and GA3C-CADRL have been introduced for D-MAMP, addressing collision avoidance, social awareness, and stochastic behavior [18]–[22]. GA3C-CADRL-NSL addresses reward sparsity by using a goal-distance-based proxy reward [22]. Agent-level representations with predefined state estimation enable access to higher-level state information, contrasting with end-to-end sensor-level approaches [22], [23].

Existing algorithms hinge on acquiring precise robot states, which remains a challenge for decentralized multi-agent motion planning (D-MAMP) based on estimated states. This work draws from methodologies in [5] for precise individual robot state estimation and proposes a Deep Reinforcement Learning-based Active Joint Localization and Target Tracking (DRL-AJLATT) framework to enhance target tracking and self-localization. The key contributions can be summarized as:

1) Reframing AJLATT as a partially observable Markov decision process suitable for reinforcement learning methods, with reward function design for estimation variance reduction.

2) A novel DRL algorithm operating on estimated robot states eliminates the need for precise states, enhancing scalability for real-world deployment.

3) Empirical evaluation in a designed scenario, demonstrating effectiveness over random and control-based methods in reducing estimation variance and mitigating collisions.

II. Preliminaries

A. Problem Formulation

In the presented scenario, a collective of denoted robots indexed as $i \in 1, 2, ..., M$ is engaged in the pursuit and monitoring of a singular designated target. At discrete time instance k, the condition of robot i (respectively, the target) is denoted as $\mathbf{x}_i^k$ (respectively, $\mathbf{x}_T^k$). $\mathbf{x}_i^k = \left[\begin{array}{c} x_i^k, y_i^k, \phi_i^k \end{array}\right]^T \in \mathbb{R}^2 \times [0, 2\pi)$, where the 2−tuple $[x_i^k, y_i^k]^T$ represents the spatial coordinates within the global reference frame, and

[1]D. Wang and W. Ren are with the Department of Electrical and Computer Engineering, University of California, Riverside, CA 92521 USA. dongming.wang@email.ucr.edu, ren@ee.ucr.edu.

[2]S. Su is with the Department of Computer Science and Engineering, State University of New York at Buffalo, NY 14260 USA. shaoshus@buffalo.edu.

[3]C. Hao is with the Department of Mechanical Engineering, University of California, Berkeley, CA 94720 USA. cehao@berkeley.edu.

This work was supported by National Science Foundation under Grant CMMI-2027139.

ϕ_i^k characterizes the angle measured in a clockwise direction from $x-$axis's positive orientation of the global reference frame to the direction in which the robot is oriented.

In this context, the robots' precise states are unknown. Following [5], the prior and posterior estimates of robot i's state are denoted as $\bar{\mathbf{x}}_i^k$ and $\hat{\mathbf{x}}_i^k$, respectively. The corresponding prior and posterior estimation errors are $\bar{\mathbf{e}}_i^k = \mathbf{x}_i^k - \bar{\mathbf{x}}_i^k$ and $\mathbf{e}_i^k = \mathbf{x}_i^k - \hat{\mathbf{x}}_i^k$, with covariances $\bar{\mathbf{p}}_i^k$ and $\hat{\mathbf{p}}_i^k$. Similarly, $\bar{\mathbf{x}}_{T_i}^k$ and $\hat{\mathbf{x}}_{T_i}^k$ denote robot i's prior and posterior estimates of the target state, with covariances $\bar{\mathbf{p}}_{T_i}^k$ and $\hat{\mathbf{p}}_{T_i}^k$.

The objective is to derive a control policy $\pi_\theta(\mathbf{a}_i^k|\mathbf{o}_i^k)$, parameterized by θ, that maps each robot's observations $\mathbf{o}_i^k$ at time k to its control inputs $\mathbf{a}_i^k$. Due to the limited communication range compared to the overall formation scale, the control is performed in a distributed manner.

B. Graphs

Communication graph G_c^k and sensing graph G_s^k are the two distinct directed graphs that are defined to represent the multi-robot system's communication and sensing structures.

The communication graph is defined as $G_c^k = (\mathcal{V}, \mathcal{E}_c^k)$, where $\mathcal{V} = \{1, 2, \ldots, M\}$ represents the set of robots and $\mathcal{E}_c^k \doteq \{(j,i)|\mathbf{x}_j^k \in \mathbf{U}(\mathbf{x}_i^k, d_c)\}$, where $(j,i) \in \mathcal{E}_c^k$ if robot i can receive information from robot j, and d_c indicates the radius of communication. Obviously, $(i,i) \in \mathcal{E}_c^k$ always holds. Furthermore, the communicating neighbor set of robot i is defined as $\mathcal{N}_{c,i}^k = \{j \neq i|(j,i) \in \mathcal{E}_c^k\}$. Similarly, the inclusive communicating neighbor set of robot i is defined as $\mathcal{I}_{c,i}^k = \{j|(j,i) \in \mathcal{E}_c^k\}$.

The sensing area of robot i is defined as a sector with the sensing radius d_s as the radius, centered on the orientation of the robot, and having a central angle of $2\theta_s$. Based on the setting of sensing area, the sensing graph is defined as $G_s^k = (\mathcal{V}, \mathcal{E}_s^k)$, where

$$\mathcal{E}_s^k \doteq \{(j,i)|\mathbf{x}_j^k \in \mathbf{U}(\mathbf{x}_i^k, d_s) \& \frac{(\mathbf{x}_j^k - \mathbf{x}_i^k)\cdot \mathbf{d}_i^k}{\|\mathbf{x}_j^k - \mathbf{x}_i^k\|} \geq \cos\theta_s\},$$

where (j,i) means robot j can be detected by i, and $\mathbf{d}_i^k = [\cos\phi_i^k, \sin\phi_i^k, 0]^T$ represents the orientation of robot i. The set of sensing neighbors of robot i is defined as $\mathcal{N}_{s,i}^k = \{j \neq i|(j,i) \in \mathcal{E}_s^k\}$.

Assuming $d_c \geq d_s$, it can be asserted that all sensing neighbors of robot i also simultaneously serve as communication neighbors of robot i.

C. Motion and Measurement Models

The motion model is considered a nonlinear dynamic model with additive noise, which can be presented as

$$\mathbf{x}_i^{k+1} = f_i(\mathbf{x}_i^k, \mathbf{a}_i^k) + \mathbf{w}_i^k, \tag{1}$$

$$\mathbf{x}_T^{k+1} = g(\mathbf{x}_T^k, \mathbf{u}_T^k) + \mathbf{w}_T^k, \tag{2}$$

where $\mathbf{a}_i^k \sim \pi(\mathbf{a}_i^k|\mathbf{o}_i^k)$ represents the control input to robot i at time k, $\mathbf{u}_T^k$ represents the control input to the target, and $\mathbf{w}_i^k \sim \mathcal{N}(0, \mathbf{Q}_i^k)$ and $\mathbf{w}_T^k \sim \mathcal{N}(0, \mathbf{Q}_T^k)$ are, respectively, the additive white Gaussian noises for robot i and the target.

At the time instant k, when the condition $j \in \mathcal{N}_{s,i}^k$ is satisfied, robot i becomes capable of acquiring a robot-to-robot measurement $\mathbf{z}_{R_{ij}}^k$. Alternatively, in the circumstance where robot j assumes the role of the target, robot i is enabled to obtain a measurement denoted as $\mathbf{z}_{R_{iT}}^k$, representing the robot-to-target interaction.The measurement models are defined as

$$\begin{aligned}\mathbf{z}_{ij}^k &= h_{ij}(\mathbf{x}_i^k, \mathbf{x}_j^k) + \mathbf{v}_{ij}^k,\\ \mathbf{z}_{T_i}^k &= h_{T_i}(\mathbf{x}_i^k, \mathbf{x}_T^k) + \mathbf{v}_{T_i}^k,\end{aligned} \tag{3}$$

where $\mathbf{v}_{ij}^k \sim \mathcal{N}(0, \mathbf{R}_{ij}^k)$ and $\mathbf{v}_{T_i}^k \sim \mathcal{N}(0, \mathbf{R}_{T_i}^k)$ are the measurement noises assumed to be white Gaussian. The measurement noises are assumed to be mutually uncorrelated across robots and uncorrelated with the process noises.

D. Joint Localization and Target Tracking

The paradigm of fully distributed JLATT [5] that is employed entails the concurrent and integrated estimation of both individual robot and target states. This method exclusively leverages intrinsic robot-derived data as well as information garnered from neighboring robots within a one-hop communication range. Of great significance is its capacity to uphold the consistency of the estimation throughout this process.

1) Propagation: The prior estimate of robot i's state and its corresponding covariance are propagated as

$$\begin{aligned}\bar{\mathbf{x}}_i^k &= f_i(\hat{\mathbf{x}}_i^{k-1}, \mathbf{a}_i^{k-1}),\\ \bar{\mathbf{p}}_i^k &= \mathbf{\Phi}_i^{k-1}\hat{\mathbf{p}}_i^{k-1}(\mathbf{\Phi}_i^{k-1})^\mathsf{T} + \bar{\mathbf{Q}}_i^{k-1}\\ &= \rho_{Rp}(\hat{\mathbf{p}}_i^{k-1}, \hat{\mathbf{x}}_i^{k-1}, \mathbf{Q}_i^{k-1}),\end{aligned} \tag{4}$$

where $\mathbf{\Phi}_i^{k-1} = \frac{\partial f_i}{\partial \mathbf{x}_i}(\hat{\mathbf{x}}_i^{k-1}, \mathbf{a}_i^{k-1}), \mathbf{G}_i^{k-1} = \frac{\partial f_i}{\partial \mathbf{w}_i}(\hat{\mathbf{x}}_i^{k-1}, \mathbf{a}_i^{k-1})$, and $\bar{\mathbf{Q}}_i^{k-1} = \mathbf{G}_i^{k-1}\mathbf{Q}_i^{k-1}(\mathbf{G}_i^{k-1})^\mathsf{T}$.

2) State Update: For the purpose of updating the state estimation for robot i, the initial step involves the derivation of correction pairs denoted as $(\mathbf{s}_{il}^k, \mathbf{y}_{il}^k)$ and $(\mathbf{s}_{T_i}^k, \mathbf{y}_{T_i}^k)$. This derivation is predicated upon the utilization of robot i's inter-robot measurements, represented as $\mathbf{z}_{il}^k$ where $l \in \mathcal{N}_{s,i}^k$, as well as its measurements concerning the robot-target relationship, encapsulated by $\mathbf{z}_{T_i}^k$. The computation of these correction pairs is articulated as follows:

$$\mathbf{s}_{il}^k = (\mathbf{H}_{il}^k)^\mathsf{T}(\bar{\mathbf{R}}_{il}^k)^{-1}\mathbf{H}_{il}^k, \tag{5a}$$

$$\mathbf{y}_{il}^k = (\mathbf{H}_{il}^k)^\mathsf{T}(\bar{\mathbf{R}}_{il}^k)^{-1}(\bar{\mathbf{z}}_{il}^k + \mathbf{H}_{il}^k\bar{\mathbf{x}}_i^k), \tag{5b}$$

where $\mathbf{H}_{il}^k = \frac{\partial \mathbf{h}_{il}}{\partial \mathbf{x}_i^k}(\bar{\mathbf{x}}_i^k, \bar{\mathbf{x}}_l^k), \bar{\mathbf{R}}_{il}^k = \mathbf{R}_{il}^k + \widetilde{\mathbf{H}}_{il}^k\bar{\mathbf{p}}_l^k(\widetilde{\mathbf{H}}_{il}^k)^\mathsf{T}, \widetilde{\mathbf{H}}_{il}^k = \frac{\partial \mathbf{h}_{il}}{\partial \mathbf{x}_l^k}(\bar{\mathbf{x}}_i^k, \bar{\mathbf{x}}_l^k)$, and $\bar{\mathbf{z}}_{il}^k = \mathbf{z}_{il}^k - \mathbf{h}_{il}(\bar{\mathbf{x}}_i^k, \bar{\mathbf{x}}_l^k)$.

Subsequently, the correction pairs are subjected to the Covariance Intersection (CI) algorithm [24], in order to calculate a consistent estimate denoted as $\breve{\mathbf{x}}_i$ along with its corresponding covariance representation, referred to as $\breve{\mathbf{p}}_i$.

$$\breve{\mathbf{p}}_i^k = (\sum_{l \in \mathcal{N}_{s,i}^k} \eta_{il}^k \mathbf{s}_{il}^k + \eta_{T_i}^k \mathbf{s}_{T_i}^k)^{-1}, \tag{6a}$$

$$\breve{\mathbf{x}}_i^k = \breve{\mathbf{p}}_i^k (\sum_{l \in \mathcal{N}_{s,i}^k} \eta_{il}^k \mathbf{y}_{R_{il}}^k + \eta_{T_i}^k \mathbf{y}_{T_i}^k), \tag{6b}$$

where $\eta_{il}^k \in [0, 1]$, and $\eta_{T_i}^k$ adheres to $\eta_{T_i}^k = 0$ if robot i cannot detect the target, and varies within [0, 1] otherwise. The condition $\sum_{l \in \mathcal{N}_{s,i}^k} \eta_{il}^k + \eta_{T_i}^k = 1$ holds true. These coefficients, η_{il} and η_{T_i}, are strategically determined to minimize the trace of covariance matrix $\breve{\mathbf{p}}_i^k$.

Utilizing the estimation pair $(\breve{\mathbf{p}}_i^k, \breve{\mathbf{x}}_i^k)$ alongside the prior estimation pair $(\bar{\mathbf{p}}_i^k, \bar{\mathbf{x}}_i^k)$ delineated earlier, the CI algorithm is employed to integrate the two sets of estimates, yielding the ensuing posterior estimation pair denoted as $(\hat{\mathbf{p}}_i^k, \hat{\mathbf{x}}_i^k)$, which is characterized by the following expression:

$$\hat{\mathbf{p}}_i^k = \left(\zeta_{i1}^k (\breve{\mathbf{p}}_i^k)^{-1} + \zeta_{i2}^k (\bar{\mathbf{p}}_i^k)^{-1} \right)^{-1}, \tag{7a}$$

$$\hat{\mathbf{x}}_i^k = \hat{\mathbf{p}}_i^k \left(\zeta_{i1}^k (\breve{\mathbf{p}}_i^k)^{-1} \breve{\mathbf{x}}_i^k + \zeta_{i2}^k (\bar{\mathbf{p}}_i^k)^{-1} \bar{\mathbf{x}}_i^k \right), \tag{7b}$$

where ζ_{i1}^k and $\zeta_{i2}^k \in [0, 1]$, subject to $\zeta_{i1}^k + \zeta_{i2}^k = 1$, can be strategically designed for minimizing the trace of $\hat{\mathbf{p}}_i^k$.

For pithy, let's focus on the propagation and state update processes for robots. A similar approach can be employed to obtain corresponding values for the target in a nearly parallel fashion. These values include $\hat{\mathbf{x}}_{T_i}^{k-1}$, $\hat{\mathbf{p}}_{T_i}^{k-1}$, $\bar{\mathbf{x}}_{T_i}^k$, $\bar{\mathbf{p}}_{T_i}^k$, as well as the pairs $(\mathbf{s}_{T_i}^k, \mathbf{y}_{T_i}^k)$ and $(\tilde{\mathbf{s}}_{T_i}^k, \tilde{\mathbf{y}}_{T_i}^k)$. For details, refer to [25].

III. Deep Reinforcement Learning-Based Active Joint Localization and Target Tracking

A. General Framework

In the scenario considered, a single robot i generates its individual control input $\mathbf{a}_i^k \sim \pi_\theta(\mathbf{a}_i^k | \mathbf{o}_i^k)$ based on its own estimation and direct communication with its one-hop neighbors $j \in \mathcal{N}_{c,i}$. This setup conforms to the structure of a Multi-Agent POMDP [16]. Furthermore, under the assumption of homogeneity among all robots, allowing for centralized training of algorithms across different robots, the algorithm falls into the category of centralized training and distributed execution. A synopsis of the DRL-AJLATT algorithm framework is presented in Table I and with details shown later.

B. Deep Reinforcement Learning Setup

This subsection will elucidate the intricacies associated with the observation space, the action space, the construction of the reward function, and the underlying justification for their specific design choices.

1) ***Observation Space:*** The observation of robot i at time instance k is defined as $\mathbf{o}_i^k = (\mathbf{o}_{i_L}^k, \mathbf{o}_{i_N}^k)$.

$$\begin{aligned}\mathbf{o}_{i_L}^k &= (\mathbf{s}_i^k, \mathbf{\Xi}_i^k, \mathbf{b}_{gi}) \\ &= \{[(\hat{\mathbf{x}}_i^k, \hat{\mathbf{p}}_i^k), (\hat{\mathbf{x}}_{T_i}^k, \hat{\mathbf{p}}_{T_i}^k)], \cup_{j \in \mathcal{N}_{s,i}^k} \mathbf{z}_{ij}^k, \mathbf{b}_{gi}\},\end{aligned}$$

TABLE I: DRL-based AJLATT by robot i

```
Initialization:
1    Initialize $\hat{\mathbf{x}}_i^0$,$\hat{\mathbf{p}}_i^0$, $\hat{\mathbf{x}}_{T_i}^0$, $\hat{\mathbf{p}}_{T_i}^0$, and set $\mathbf{a}_i^{-1} = \mathbf{0}$.

At time $k-1$
Information Exchange for AJLATT:
2.1  Send $\mathbf{o}_{iL}^{k-1}$ to robot $j$, $i \in \mathcal{N}_{c,j}^{k-1}$.
2.2  Receive $\mathbf{o}_{jL}^{k-1}$ from robot $j \in \mathcal{N}_{c,i}^{k-1}$.
AJLATT Motion Planning:
3    Generate $\mathbf{a}_i^{k-1} \sim \pi_\theta(\mathbf{a}_i^{k-1} | \mathbf{o}_i^{k-1})$.
4    With $\mathbf{a}_i^{k-1}$, $\hat{\mathbf{x}}_i^{k-1}$ and $\hat{\mathbf{p}}_i^{k-1}$, propagate robot $i$'s estimate
     of its own state $\bar{\mathbf{x}}_i^k$, $\bar{\mathbf{p}}_i^k$; with known $\mathbf{u}_T^k$, $\hat{\mathbf{x}}_{T_i}^{k-1}$ and $\hat{\mathbf{p}}_{T_i}^{k-1}$,
     obtain $\bar{\mathbf{x}}_{T_i}^k$ and $\bar{\mathbf{p}}_{T_i}^k$.
At time $k$
Update:
5.1  Obtain the robot-robot measurements $\mathbf{z}_{il}^k$, $l \in \mathcal{N}_{s,i}^k$, and
     robot-target measurement $\mathbf{z}_{T_i}^k$ (if the target is detected by
     robot $i$) and generate the correction pairs $(\mathbf{s}_{il}^k, \mathbf{y}_{il}^k)$,
     $(\mathbf{s}_{T_i}^k, \mathbf{y}_{T_i}^k)$, $(\tilde{\mathbf{s}}_{T_i}^k, \tilde{\mathbf{y}}_{T_i}^k)$.
5.2  Send $(\tilde{\mathbf{s}}_{T_i}^k, \tilde{\mathbf{y}}_{T_i}^k)$, $(\bar{\mathbf{p}}_{T_i}^k, \bar{\mathbf{x}}_{T_i}^k)$ to robot $j$, $i \in \mathcal{N}_{c,j}^k$.
5.3  Receive $(\tilde{\mathbf{s}}_{T_j}^k, \tilde{\mathbf{y}}_{T_j}^k)$, $(\bar{\mathbf{p}}_{T_j}^k, \bar{\mathbf{x}}_{T_j}^k)$ from robot $j$, $j \in \mathcal{N}_{c,i}^k$.
5.4  Calculate the posterior robot estimate pair $(\hat{\mathbf{p}}_i^k, \hat{\mathbf{x}}_i^k)$, and
     the posterior target estimate pair $(\hat{\mathbf{p}}_{T_i}^k, \hat{\mathbf{x}}_{T_i}^k)$.
Applicable When Training
Policy Update:
6.1  Operate steps 2-5 until truncated, get trajectories of robots.
6.2  Partition the trajectory of each robot into discrete segments
     with fixed length T. Save the segments into replay buffer.
6.3  if there are enough samples: go to step 6.5.
     else go to step 6.4.
6.4  Reset all robots to their initials, and go back to step 6.1.
6.5  Update policy $\pi_\theta(\mathbf{a}_i^{k-1} | \mathbf{o}_i^{k-1})$ by PPO algorithm [26].
     Move on to the next episode until the end.
```

where $\mathbf{s}_i^k = [(\hat{\mathbf{x}}_i^k, \hat{\mathbf{p}}_i^k), (\hat{\mathbf{x}}_{T_i}^k, \hat{\mathbf{p}}_{T_i}^k)]$ denotes the state of robot i, $\mathbf{\Xi}_i^k$ represents robot i's measurement concerning its sensing neighbors, and $\mathbf{b}_{gi} = (x_{io}^k, y_{io}^k)$ indicates the vector originating from robot i to the nearest boundary of the map within its sensing area. $\mathbf{o}_{i_N}^k = \bigcup_{j \in \mathcal{N}_{c,i}^k} \mathbf{s}_j^k$ conveys the information that is reachable through the one-hop neighbors of robot i.

To handle varying dimensionality of robot observations $\mathbf{o}_i^k$ based on the number of sensed and communicable robots, constraints limit sensing to maximum k_s robots and communication to maximum k_c robots. If exceeded, random selection resolves redundancy; if lower, zero-padding maintains consistent dimensionality.

2) ***Action Space:*** The robots are directly controlled by adjusting their linear and angular velocities. The strategy involves a continuous action space formulation that encompasses both linear and angular velocities, which is

$$\mathbf{a}_i^k = (v_i^k, \omega_i^k) \in [v_{min}, v_{max}] \times [\omega_{min}, \omega_{max}] = \mathcal{R}_{control},$$

where $[v_{min}, v_{max}]$ and $[\omega_{min}, \omega_{max}]$ denote the interval of linear and angular velocity respectively.

The policy π_θ is represented with a Gaussian distribution $\mathbf{f}_i^k \sim \pi_\theta(\mathbf{f}_i^k | \mathbf{o}_i^k) \doteq \mathcal{N}(\mu_a, \sigma^2)$, where $\mu_a \in \mathbb{R}^2$ denotes a function originating from the robot's observation at time step k, characterized by a feed-forward neural network.

Meanwhile, the variance, denoted as $\sigma^2 \in \mathbb{R}$, constitutes a singular learned parameter that remains uncorrelated with the system's state. When considering the vector $\mathbf{f}_i^k$ across the defined control interval, the application of a clipping procedure becomes operative, wherein $\mathbf{a}_i^k = \text{clip}(\mathbf{f}_i^k, \mathcal{R}_{control})$, thereby ensuring its adherence to the bounds stipulated by $\mathcal{R}_{control}$. For the sake of parsimony, the policy is represented as $\pi_\theta(\mathbf{a}_i^k|\mathbf{o}_i^k)$.

*3) **Reward Design:*** The reward function consists of four major components: $r_i^k = r_{Ti}^k + r_{C_Ti}^k + r_{Ei}^k + r_{C_Oi}^k$, where $r_{Ti}^k = -\alpha(\text{tr}(\hat{P}_{T_i}) - trS)$ bestows rewards for improving robots' target estimation precision, with the lower limit for the trace of the covariance matrix set as $\text{tr}(S)$.

$$r_{C_Ti}^k = \begin{cases} -\gamma \sum_{j \in \mathcal{N}_{c,i}^k} \frac{\cos(\frac{\pi \|\hat{\mathbf{x}}_i^k - \hat{\mathbf{x}}_j^k\|}{2d_{co}})}{\sin(\frac{\pi \|\hat{\mathbf{x}}_i^k - \hat{\mathbf{x}}_j^k\|}{2d_{co}}) + \epsilon_c}, \ \|\hat{\mathbf{x}}_i^k - \hat{\mathbf{x}}_j^k\| \leq 2d_{co} \\ 0, \text{otherwise} \end{cases}$$

rewards the robot for avoiding collision, where d_{co} is the allowed lower bound between robots and ϵ_c is a positive constant for preventing the denominator from being too small. $r_{Ei}^k = -\epsilon \left(\text{tr}\, \hat{P}_i - \text{tr}\, S\right)$ rewards the robot for better estimating the state of itself. $r_{C_Oi}^k = -\eta\, (\text{if} : \|\mathbf{b}_{gi}\| < d_{co})$ confers rewards upon the robot as a recognition of its proactive avoidance of collisions with the boundaries of the map.

C. Attention-based Embedding

To elucidate the relative significance attributed to individual neighbors, the attention mechanism [27], is introduced as a pivotal construct. Inspired by the attention framework expounded in [28], a tailored modification is formulated to effectively accommodate the specific scenario delineated therein. This can be articulated as $e_j^k = \psi_e(\mathbf{o}_{iN}^k), e_m^k = \frac{1}{k_c}\sum_{j=1}^{k_c} e_j, \alpha_j^k = \psi_\alpha(e_j^k, e_m^k)$, where e_j signifies the embedding vectors attributed to individual neighbors, e_m embodies the mean value computed across the entirety of the neighborhood, and ψ_e along with ψ_α denote fully-connected neural networks. Through the employment of the softmax operation applied to α_j for the calculation of attention scores, ensuring their cumulative sum equates to unity, the resulting outcome manifests as the neighborhood embedding: $e_\eta^k = \sum_{j=1}^{k_c} \text{Softmax}(\alpha_j^k)\psi_h(e_j^k)$, where ψ_h represents an additional hidden layer. Preceding the computation of the distribution mean μ_a, an additional fully-connected neural network ϕ_s is introduced to encapsulate the embedding of the state pertaining to robot i itself: $e_s^k = \phi_s(\mathbf{o}_{iL}^k)$. Subsequently, the distribution mean μ_a can be derived by combining the information from these two embeddings: $\mu_a^k = \phi_a(e_s^k, e_\eta^k)$, where ϕ_a is also a fully-connected neural network.

D. Training Algorithm

The Proximal Policy Optimization (PPO) algorithm is employed for the purpose of identifying the optimal policy. In the following contents of this section, the time indices and robot identity are omitted for simplicity where possible.

Generally, there are two variants of PPO, PPO-clip and PPO-penalty. Comparing PPO-clip and PPO-penalty, the advantage of PPO-clip lies in its simpler and more stable approach to policy optimization. PPO-clip directly limits policy changes through clipping, leading to controlled updates and enhanced convergence. PPO-penalty introduces complexity with penalty terms and coefficients, potentially leading to less stable training dynamics. Therefore, PPO-clip is preferred for its effectiveness and ease of implementation [26], that is the reason why PPO-clip is employed for training in this paper.

IV. SIMULATION

We substantiate the enhanced performance of our Deep Reinforcement Learning (DRL)-based approach in collision avoidance and target tracking through a comparative analysis with two baseline methods: one relying on random motion and the other on control-based strategies [25].

A. Simulation Setup

Consider the scenarios where $M = 6$ robots and a target move on a 2-dimension surface. The unicycle model is adopted for robots and the target in our simulation. The motion models (1) and (2) can be expressed as

$$\begin{aligned} x_i^k &= x_i^{k-1} + v_i^{k-1}\delta t \cos(\phi_i^{k-1}) + w_{v_i}^{k-1}, \\ y_i^k &= y_i^{k-1} + v_i^{k-1}\delta t \sin(\phi_i^{k-1}) + w_{v_i}^{k-1}, \\ \phi_i^k &= \phi_i^{k-1} + \omega_i^{k-1}\delta t + w_{\omega_i}^{k-1}, \end{aligned} \tag{8}$$

where $i \in \{1, \dots, M\} \cup \{T\}$, $\delta t = 0.5$ s is the sampling interval, and $\mathbf{w}_i = [w_{v_i}^{k-1}, w_{\omega_i}^{k-1}]^T$ represents process noises for the linear and angular velocities.

For robot i, the input $\mathbf{a}_i^{k-1}$ is generated by policy $\pi_\theta(\mathbf{a}_i^k|\mathbf{o}_i^k)$ trained by our DRL-based AJLATT algorithm. The limitations that we set for v_i and ω_i are $v_i \in [0, 1]$ m/s, and $\omega_i \in [-\frac{\pi}{4}, \frac{\pi}{4}]$ rad/s. The target's input $\mathbf{u}_T^k = [v_T^k, \omega_T^k]^T$ is set in advance. The standard deviations for $w_{v_i}^{k-1}$ and $w_{\omega_i}^{k-1}$ for each robot i, $i \in \{1, \dots, M\}$, are, respectively, $\sigma_{v_i}^{k-1} = 0.1$ m/s and $\sigma_{\omega_i}^{k-1} = 0.5°$, and $\sigma_{v_T}^{k-1} = 0.2$ m/s, $\sigma_{\omega_T}^{k-1} = 0.5°$ for the target. Each robot has a 5m's radius of communication range, and a 4m's radius of field of view with $\theta_s = \frac{\pi}{4}$.

The relative distance-bearing measurement model is adopted for each robot in our simulation. If robot i detects robot j at time step k, then the relative measurement is given by

$$\mathbf{z}_{ij}^k = \begin{bmatrix} \sqrt{(x_j^k - x_i^k)^2 + (y_j^k - y_i^k)^2} \\ \text{atan2}((y_j^k - y_i^k), (x_j^k - x_i^k)) - \theta_i^k \end{bmatrix} + \mathbf{v}_{ij}^k.$$

The standard deviation of the distance noise is set to 0.01 m, and the standard deviation of the bearing noise is set to $2°$. The same measurement model is used for the robot-to-target measurement $\mathbf{z}_{T_i}$ that we will use in the scenarios to be introduced later.

The true states of robots and target being unavailable, each estimates its initial state $\hat{\mathbf{x}}_i^0$ from a normal distribution with mean as the true initial state $\mathbf{x}_i^0$ and covariance $\hat{\mathbf{p}}_i^0$. The reinforcement network uses actor and critic networks with 2 fully connected hidden layers (512 and 256 neuron units), batch size 32, replay buffer size 10^6, and 300 training episodes.

Three different motion strategies[1] are compared:

1) *Random Motion (RM)*: use the same settings as in [25].
2) *Control-based AJLATT Algorithm (C-AJLATT) [25].*
3) *DRL-based AJLATT (DRL-AJLATT)*: the algorithm proposed in this work.

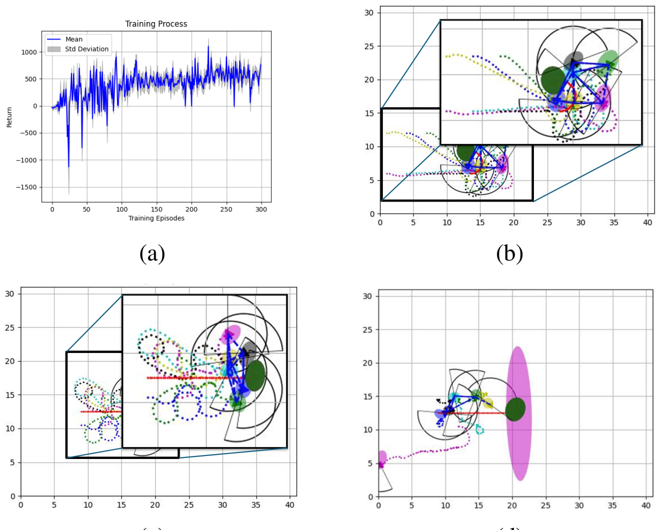

Fig. 1: (a) shows training convergence around 150 episodes. (b) shows robot formation trajectory using DRL-AJLATT, with red as target path. (c) shows trajectory with C-AJLATT. (d) shows trajectory with RM algorithm.

TABLE II: Performance comparison for different methods. In all three aspects, DRL-AJLATT demonstrates superior performance.

Metrics	Method	Empty Environment
Reward	RM	-979.85
	C-AJLATT	563.08
	DRL-AJLATT	**691.51**
Robot Covariance	RM	4.512
	C-AJLATT	2.557
	DRL-AJLATT	**1.953**
Target Covariance	RM	6.055
	C-AJLATT	2.626
	DRL-AJLATT	**2.381**

B. Simulation Metrics

In order to compare the performance of our algorithm with other motion strategies,[2] we choose the following metrics to evaluate the performance.

[1]As the optimization-based algorithm in [25] uses exhaustive grid search, which is extremely time consuming and less realistic, the algorithm is not adapted for comparison here.

[2]As [29] has a different focus, which does not consider self localization and collision avoidance, there is no basis for comparison with it.

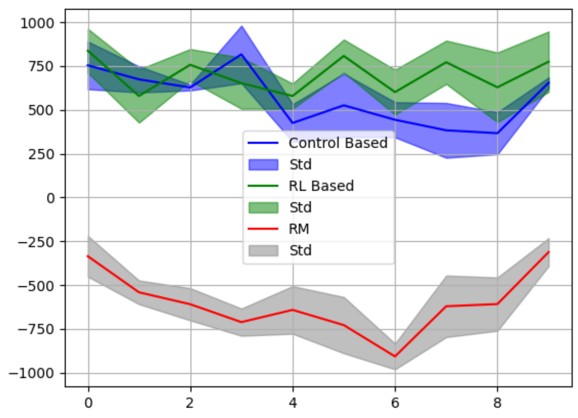

Fig. 2: Upon conducting ten runs of the trained DRL-AJLATT, along with C-AJLATT and RM algorithms, each executed with six distinct random seeds.

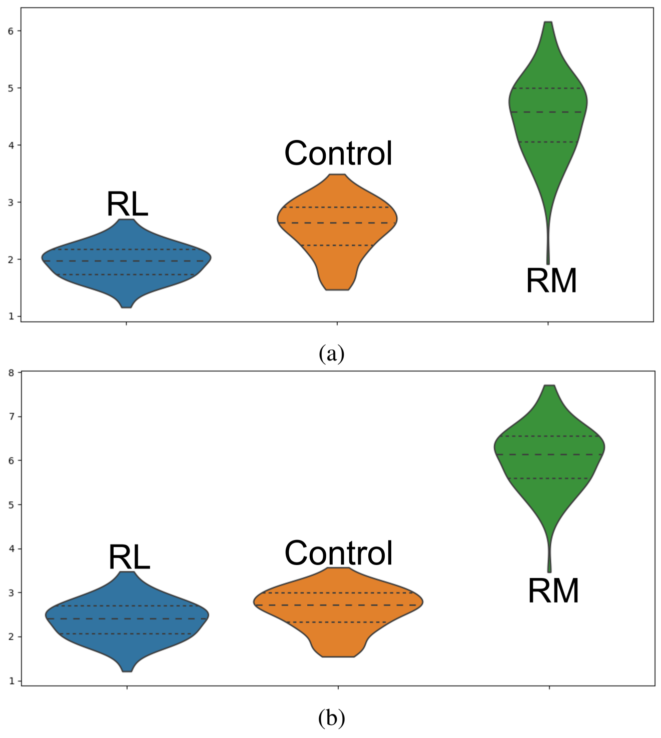

Fig. 3: Subfigure (a) illustrates robot covariance, and subfigure (b) depicts target covariance. Both figures highlight DRL-AJLATT's notable superiority in reducing estimation variance compared to other methods.

- *Reward:* We assess robot performance using the term 'Reward,' which indicates their success in achieving the desired goal. To facilitate comparison, we compute the rewards for C-AJLATT and RM algorithms in a manner consistent with how we evaluate DRL-AJLATT
- *Trace of Covariance:* The sum of the eigenvalues of the covariance matrix measures the uncertainty in estimates of robot and target states. The covariance of each robot's

self-estimate and target estimate is for comparison.

C. Results

In Figure 1, a comparison between subfigures (b) and (c) reveals that, despite both groups of robots being guided by DRL-AJLATT and C-AJLATT successfully capturing the target, the C-AJLATT algorithm displays less efficient movement. Conversely, the robots guided by the DRL-AJLATT algorithm exhibit more effective tracking of the target. C-AJLATT-guided robots exhibit unnecessary movements, such as spinning. DRL-AJLATT-guided robots are more efficient, moving directly towards the target. Subfigure (d) illustrates that RM is the least effective in this context.Table II demonstrates DRL-AJLATT's consistent superiority in all aspects. Figure 3 confirms DRL-AJLATT's better performance in reducing estimation covariance compared to C-AJLATT and RM. In summary, DRL-AJLATT significantly outperforms RM and C-AJLATT algorithms in the AJLATT task of this paper.

V. Conclusion

We introduce a DRL-based algorithm for the AJLATT problem using a team of robots. Simulations validate the efficacy of our DRL-based approach for AJLATT tasks. Our algorithm relies on estimated robot states, enhancing scalability for real-world applications. However, limitations include training in a fixed environment, a lack of theoretical grounding, and underutilization of the estimation model. Future work aims to evaluate generalizations through real-world experiments and explore relevant theories.

Acknowledgment

We wish to express our gratitude to Zhiqiang Jian, Mingxiao Huo, and Xinghao Zhu for their invaluable discussions.

References

[1] G. Huang, M. Kaess, and J. J. Leonard, "Consistent unscented incremental smoothing for multi-robot cooperative target tracking," *Robotics and Autonomous Systems*, vol. 69, pp. 52–67, 2015.

[2] A. Ahmad, G. D. Tipaldi, P. Lima, and W. Burgard, "Cooperative robot localization and target tracking based on least squares minimization," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2013, pp. 5696–5701.

[3] F. M. Mirzaei, A. I. Mourikis, and S. I. Roumeliotis, "On the performance of multi-robot target tracking," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2007, pp. 3482–3489.

[4] N. Atanasov, R. Tron, V. M. Preciado, and G. J. Pappas, "Joint estimation and localization in sensor networks," in *Proceedings of the IEEE Conference on Decision and Control*, 2014, pp. 6875–6882.

[5] P. Zhu and W. Ren, "Fully distributed joint localization and target tracking with mobile robot networks," *IEEE Transactions on Control Systems Technology*, vol. 29, no. 4, pp. 1519–1532, 2020.

[6] Y. Cao and W. Ren, "Distributed coordinated tracking with reduced interaction via a variable structure approach," *IEEE Transactions on Automatic Control*, vol. 57, no. 1, pp. 33–48, 2011.

[7] B. Charrow, N. Michael, and V. Kumar, "Cooperative multi-robot estimation and control for radio source localization," *The International Journal of Robotics Research*, vol. 33, no. 4, pp. 569–580, 2014.

[8] B. Schlotfeldt, D. Thakur, N. Atanasov, V. Kumar, and G. J. Pappas, "Anytime planning for decentralized multirobot active information gathering," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 1025–1032, 2018.

[9] K. Zhou and S. I. Roumeliotis, "Multirobot active target tracking with combinations of relative observations," *IEEE Transactions on Robotics*, vol. 27, no. 4, pp. 678–695, 2011.

[10] N. Atanasov, J. Le Ny, K. Daniilidis, and G. J. Pappas, "Information acquisition with sensing robots: Algorithms and error bounds," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 6447–6454.

[11] A. W. Stroupe and T. Balch, "Value-based action selection for observation with robot teams using probabilistic techniques," *Robotics and Autonomous Systems*, vol. 50, no. 2-3, pp. 85–97, 2005.

[12] K. Hausman, J. Müller, A. Hariharan, N. Ayanian, and G. S. Sukhatme, "Cooperative multi-robot control for target tracking with onboard sensing," *The International Journal of Robotics Research*, vol. 34, no. 13, pp. 1660–1677, 2015.

[13] F. Morbidi and G. L. Mariottini, "Active target tracking and cooperative localization for teams of aerial vehicles," *IEEE Transactions on Control Systems Technology*, vol. 21, no. 5, pp. 1694–1707, 2013.

[14] F. Meyer, H. Wymeersch, M. Fröhle, and F. Hlawatsch, "Distributed estimation with information-seeking control in agent networks," *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 11, pp. 2439–2456, 2015.

[15] A. Tampuu, T. Matiisen, D. Kodelja, I. Kuzovkin, K. Korjus, J. Aru, J. Aru, and R. Vicente, "Multiagent cooperation and competition with deep reinforcement learning," *PloS one*, vol. 12, no. 4, p. e0172395, 2017.

[16] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," *arXiv preprint arXiv:1706.02275*, 2017.

[17] T. Fan, P. Long, W. Liu, and J. Pan, "Distributed multi-robot collision avoidance via deep reinforcement learning for navigation in complex scenarios," *The International Journal of Robotics Research*, vol. 39, no. 7, pp. 856–892, 2020.

[18] M. Everett, Y. F. Chen, and J. P. How, "Motion planning among dynamic, decision-making agents with deep reinforcement learning," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 3052–3059.

[19] Y. F. Chen, M. Liu, M. Everett, and J. P. How, "Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 285–292.

[20] Y. F. Chen, M. Everett, M. Liu, and J. P. How, "Socially aware motion planning with deep reinforcement learning," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 1343–1350.

[21] M. Everett, Y. F. Chen, and J. P. How, "Collision avoidance in pedestrian-rich environments with deep reinforcement learning," *IEEE Access*, vol. 9, pp. 10 357–10 377, 2021.

[22] S. H. Semnani, H. Liu, M. Everett, A. De Ruiter, and J. P. How, "Multi-agent motion planning for dense and dynamic environments via deep reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3221–3226, 2020.

[23] L. Tai, G. Paolo, and M. Liu, "Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 31–36.

[24] S. Julier and J. K. Uhlmann, "General decentralized data fusion with covariance intersection," in *Handbook of multisensor data fusion*. CRC Press, 2017, pp. 339–364.

[25] S. Su, P. Zhu, and W. Ren, "Multirobot fully distributed active joint localization and target tracking," *IEEE Transactions on Control Systems Technology*, 2023.

[26] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[27] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[28] S. Batra, Z. Huang, A. Petrenko, T. Kumar, A. Molchanov, and G. S. Sukhatme, "Decentralized control of quadrotor swarms with end-to-end deep reinforcement learning," in *Conference on Robot Learning*. PMLR, 2022, pp. 576–586.

[29] C. D. Hsu, H. Jeong, G. J. Pappas, and P. Chaudhari, "Scalable reinforcement learning policies for multi-agent control," *arXiv preprint arXiv:2011.08055*, 2020.