Policy Learning for Active Target Tracking over Continuous $SE(3)$ Trajectories

Pengzhi Yang PEYANG@UCSD.EDU
Shumon Koga SKOGA@UCSD.EDU
Arash Asgharivaskasi AASGHARI@UCSD.EDU
Nikolay Atanasov NATANASOV@UCSD.EDU
Department of Electrical and Computer Engineering, University of California San Diego, La Jolla, CA 92093

Editors: N. Matni, M. Morari, G. J. Pappas

Abstract

This paper develops a *model-based policy gradient algorithm* for tracking dynamic targets using a mobile agent equipped with an onboard sensor with limited field of view. The task is to obtain a continuous control policy for the mobile agent to collect sensor measurements that reduce uncertainty in the target states, measured by the target distribution entropy. We design a neural network control policy with the agent $SE(3)$ pose and the mean vector and information matrix of the joint target distribution as inputs and attention layers to handle variable numbers of targets. We also derive the gradient of the target entropy with respect to the network parameters explicitly, allowing efficient model-based policy gradient optimization.

Keywords: Active target tracking, model-based reinforcement learning, SLAM

Supplementary Material

Open-source implementation: github.com/ExistentialRobotics/RL_Active_Multi_Target_Tracking

1. Introduction

Active target tracking is a problem in which the trajectory of a sensing agent is planned to reduce uncertainty in the state of a dynamic target of interest. This problem is motivated by several applications, including search and rescue (Kumar et al., 2004), security and surveillance (Grocholsky et al., 2006), wildfire detection (Julian and Kochenderfer, 2019), and pursuit evasion (Chung et al., 2011). Active information gathering in Simultaneous Localization and Mapping (SLAM) (Cadena et al., 2016; Placed et al., 2023) is an example of active target tracking in which the target is the (static) map of the environment. The challenge of the general active target tracking problem is inherent in predicting the future target state, optimizing the sensing agent trajectory with a limited Field of View (FoV), and taking into account the stochasticity of the target motion and sensor observations.

While the general active target tracking problem is posed as a stochastic optimal control problem due to the probabilistic inference of the target states, some earlier works have reduced this complexity. Under the assumption of linear Gaussian target motion and sensor observation models, active target tracking with an information-theoretic cost results in a deterministic optimal control problem, as shown in Le Ny and Pappas (2009). Atanasov et al. (2014) proposed a computationally efficient non-myopic planning approach with a strong performance guarantee even under a long planning

horizon. Schlotfeldt et al. (2019) developed a consistent heuristic for applying A* search to the active information acquisition by deriving maximum upper bounds for the information measure. A multi-agent multi-target formulation of active information acquisition and associated scalable algorithms were studied by Atanasov et al. (2015); Schlotfeldt et al. (2018); Kantaros et al. (2019); Cai et al. (2021). While those works consider planning over discrete control space, Koga et al. (2021) proposed "iterative Covariance Regulation" (iCR), which optimizes the sensing trajectory over continuous $SE(3)$ space by deriving an analytical gradient of the cost with respect to the multi-step control sequence. Extensions of the work to occlusion-aware planning and to active SLAM under uncertain agent state was developed by Asgharivaskasi et al. (2022) and Koga et al. (2022), respectively. However, all the aforementioned works compute control inputs for a given environment and cannot be applied to a new environment without replanning.

Learning a control policy from training data obtained over several environments has been studied in the context of reinforcement learning (RL) (Sutton and Barto, 2018). RL methods employing deep neural network representations of the policy and value functions have been developed for both discrete control spaces (Mnih et al., 2015) applied to games and continuous control spaces (Lillicrap et al., 2015; Schulman et al., 2017) applied to robotics tasks. Learning a policy for active target tracking was proposed by Jeong et al. (2019) using Q-learning to maximize the mutual information between the sensor data and the target states. Hsu et al. (2021) developed a multi-agent version of Jeong et al. (2019) by incorporating an attention-block in the Q-network architecture. In addition, Tang and Ha (2021) also leveraged an attention mechanism to achieve permutation-invariance in multi-agent settings. Chen et al. (2020) focused on active landmark mapping using a graph neural network representing the exploration policy, which is trained by Q-learning within a framework of Expectation Maximization (Wang and Englot, 2020). Chaplot et al. (2020) proposed a modular and hierarchical approach to obtain a local policy by imitation learning from analytical path planners with a learned SLAM module and a global policy to maximize area coverage. Lodel et al. (2022) applied PPO (Schulman et al., 2017) for learning an information-theoretic active mapping policy to acquire reference viewpoints that maximize reward with local sensing of obstacles and the agent position. Yang et al. (2023) proposed a continuous trajectory learning method for active perception to localize multiple static landmarks, utilizing differentiable field of view for reward shaping and an attention-based neural network architecture. Learning low-level continuous control (e.g., velocity or torque) for $SE(3)$ agent kinematics using model-free RL methods is challenging because obtaining stable policy convergence requires a sufficiently large amount of experience, especially for complicated tasks.

Utilizing a known or predicted state transition model in learning algorithms can significantly reduce the required amount of samples and computation relative to model-free RL methods. Levine and Koltun (2013) developed a guided policy search that optimizes the system trajectory associated with the model by Differential Dynamic Programming (DDP) to achieve direct policy learning in control. Several variants and extensions of guided policy search were proposed by Levine and Abbeel (2014) for policy learning with unknown dynamics and by Levine et al. (2016) to obtain an end-to-end policy from visual sensing to robot action. Luo et al. (2019) incorporated a force and torque model into RL to enable high-precision robot manipulation tasks. A recent comprehensive review of the model-based RL was presented by Janner et al. (2019).

Contributions: The contributions of the paper are summarized as follows.

- We develop a novel model-based policy gradient algorithm for tracking multiple dynamic targets over continuous $SE(3)$ trajectories. A differentiable field-of-view (FoV) formulation is incorporated to enable offline learning for sensor models with limited FoV.
- We design a neural network policy architecture with an attention block to handle multiple targets and with padding and masking to enable learning over a varying number of targets during training.

2. Problem Statement

Consider an agent with pose $T_k \in SE(3) \subset \mathbb{R}^{4\times4}$ at time $t_k \in \mathbb{R}_+$, where $\{t_k\}_{k=0}^K$ for some $K \in \mathbb{N}$ is an increasing sequence. The definition of pose and its discrete-time kinematic model are:

$$T_k := \begin{bmatrix} R_k & \mathbf{x}_k \\ \mathbf{0}_{3\times1}^\top & 1 \end{bmatrix}, \quad T_{k+1} = T_k \exp\left(\tau_k \hat{\mathbf{u}}_k\right), \tag{1}$$

where $\mathbf{x}_k \in \mathbb{R}^3$ is position, $R_k \in SO(3) \subset \mathbb{R}^{3\times3}$ is orientation, $\tau_k := t_{k+1} - t_k > 0$ is the sampling-time interval, and $\mathbf{u}_k = [\mathbf{v}_k^\top, \boldsymbol{\omega}_k^\top]^\top \in \mathbb{R}^6$ is a control input, consisting of linear velocity $\mathbf{v}_k \in \mathbb{R}^3$ and angular velocity $\boldsymbol{\omega}_k \in \mathbb{R}^3$. The hat operator $\hat{(\cdot)} : \mathbb{R}^6 \to se(3)$ maps vectors in $\mathbb{R}^6$ to the Lie algebra $se(3)$ associated with the $SE(3)$ Lie group (Barfoot, 2017).

We consider a finite number of moving targets $\mathbf{y}_k = [\mathbf{y}_k^{(1)}, \ldots, \mathbf{y}_k^{(n_l)}]$, where $\mathbf{y}_k^{(j)} \in \mathbb{R}^{n_y}$ for $j \in \{1, \ldots, n_l\}$ denotes the n_y-dimensional state of j-th target at time k and n_l is the total number of targets. We assume that each target has homogeneous dynamics governed by a linear Gaussian process:

$$\mathbf{y}_{k+1}^{(j)} = A\mathbf{y}_k^{(j)} + B\boldsymbol{\xi}_k^{(j)} + \mathbf{w}_k^{(j)}, \tag{2}$$

where $A : \mathbb{R}^{n_y\times n_y}$ and $B : \mathbb{R}^{n_y\times m_y}$ are the system matrices, $\boldsymbol{\xi}_k^{(j)} \in \mathbb{R}^{m_y}$ is a known target input, and $\mathbf{w}_k^{(j)} \sim \mathcal{N}(0, W_k)$ is a stochastic process noise assumed to be Gaussian with zero mean and covariance $W_k \in \mathbb{R}^{n_y\times n_y}$.

The agent is equipped with an onboard sensor for tracking the target states. Let $\mathcal{F} \subset \mathbb{R}^3$ represent the FoV of the sensor within the agent's body frame. The set of target indices within the FoV is:

$$\mathcal{I}_\mathcal{F}(T, \{\mathbf{y}^{(j)}\}) = \left\{ j \in \{1, \ldots, n_l\} \mid \mathbf{q}\left(T, \boldsymbol{\zeta}(\mathbf{y}^{(j)})\right) \in \mathcal{F} \right\}, \tag{3}$$

where $\boldsymbol{\zeta} : \mathbb{R}^{n_y} \to \mathbb{R}^3$ transforms the target state to the 3-D coordinate of the target's location, and $\mathbf{q} : SE(3) \times \mathbb{R}^3 \to \mathbb{R}^3$ returns the agent-body-frame coordinates of $\boldsymbol{\zeta} \in \mathbb{R}^3$ given by

$$\mathbf{q}(T, \boldsymbol{\zeta}) = QT^{-1}\underline{\boldsymbol{\zeta}}, \tag{4}$$

where the projection matrix Q and the homogeneous coordinates $\underline{\boldsymbol{\zeta}}$ are defined as:

$$Q = \begin{bmatrix} I_3 & \mathbf{0}_{3\times1} \end{bmatrix} \in \mathbb{R}^{3\times4}, \quad \underline{\boldsymbol{\zeta}} = \begin{bmatrix} \boldsymbol{\zeta} \\ 1 \end{bmatrix} \in \mathbb{R}^4. \tag{5}$$

Then, a sensor measurement is denoted by $\mathbf{z}_k = [\{\mathbf{z}_k^{(j)}\}_{j\in\mathcal{I}_\mathcal{F}(T_k,\{\mathbf{y}_k^{(j)}\})}] \in \mathbb{R}^{n_z|\mathcal{I}_\mathcal{F}(T_k,\{\mathbf{y}_k^{(j)}\})|}$ where $\mathbf{z}_k^{(j)} \in \mathbb{R}^{n_z}$ is an observation of j-th target with model:

$$\mathbf{z}_k^{(j)} = H\mathbf{y}_k^{(j)} + \boldsymbol{\eta}_k, \quad \boldsymbol{\eta}_k \sim \mathcal{N}(0, V), \tag{6}$$

for all $j \in \mathcal{I}_\mathcal{F}(T_k, \{\mathbf{y}_k^{(j)}\})$, where the matrix $H \in \mathbb{R}^{n_z\times n_y}$ is the sensor matrix and $V \in \mathbb{R}^{n_z\times n_z}$ is the sensing noise covariance.

Our task is to develop a control policy for the agent to minimize uncertainty about the multiple targets using information acquired from the onboard sensor. We consider minimizing the differential entropy [1] $\mathbb{H}(\mathbf{y}_K|\mathbf{z}_{0:K}, T_{0:K})$ of the terminal target state $\mathbf{y}_K$ given a sequence of sensor observations $\mathbf{z}_{0:K}$ and the agent trajectory $T_{0:K}$. Since each target state is independent of all other target states due to the independent motion model in (2), the problem is equivalent to

$$\min \sum_{j=1}^{n_l} \mathbb{H}(\mathbf{y}_K^{(j)}|\mathbf{z}_{0:K}, T_{0:K}). \tag{7}$$

Under the Gaussian target state obeying (2) with the linear Gaussian sensor model (6), the problem (7) is equivalent to

$$\max \sum_{j=1}^{n_l} \log\det\left(Y_K^{(j)}\right), \tag{8}$$

where $Y_K^{(j)}$ is the terminal information matrix of the posterior distribution of target state $\mathbf{y}_K^{(j)}$. More precisely, we denote the prior and posterior distributions of the target state given a history of measurements as:

$$\mathbf{y}_k^{(j)}|\mathbf{z}_{0:k-1} \sim \mathcal{N}(\mathbf{p}_k^{(j)}, (P_k^{(j)})^{-1}), \quad \mathbf{y}_k^{(j)}|\mathbf{z}_{0:k} \sim \mathcal{N}(\boldsymbol{\mu}_k^{(j)}, (Y_k^{(j)})^{-1}), \tag{9}$$

for all $j \in \{1, \ldots, n_l\}$ and $k \in \{1, \ldots, K\}$. The mean and covariance (or information) matrix are updated based on the Kalman Filter, which is given by the following prediction and update steps (Atanasov et al., 2014) (here we omit the superscripts $^{(j)}$ to ease the notation but the variables are for each j-th target):

Prediction (for all $j \in \{1, \ldots, n_l\}$):
$$\mathbf{p}_{k+1} = A\boldsymbol{\mu}_k + B\boldsymbol{\xi}_k, \tag{10}$$
$$P_{k+1} = (AY_k^{-1}A^\top + W_k)^{-1}, \tag{11}$$

Update (for $j \in \mathcal{I}_\mathcal{F}(T_{k+1}, \{\mathbf{y}_{k+1}^{(j)}\})$):
$$\boldsymbol{\mu}_{k+1} = \mathbf{p}_{k+1} + K_{k+1}(\mathbf{z}_{k+1} - H\mathbf{p}_{k+1}), \tag{12}$$
$$Y_{k+1} = P_{k+1} + H^\top V^{-1}H, \tag{13}$$
$$K_{k+1} = P_{k+1}^{-1}H_{k+1}^\top(H_{k+1}P_{k+1}^{-1}H_{k+1}^\top + V_{k+1})^{-1}. \tag{14}$$

However, since the update step is performed only for targets within FoV, which is known only after obtaining the sensing with limited FoV, the implementation above is not possible in the offline planning stage. To enable planning before measurements are obtained, following Koga et al. (2021),

1. The differential entropy of a continuous random variable Y with probability density function p is defined as $\mathbb{H}(Y) := -\int p(y)\log p(y)dy$.

we introduce a differentiable FoV formulation to relax the index condition and enable gradient computation. Moreover, during training, we suppose that the sensor noise is negligible to enable offline non-myopic planning without acquiring measurements, thereby leading to identical prior and posterior means. We then design the control policy $\mathbf{u}_k = \boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{s}_k)$ as a deep neural network, where $\mathbf{s}_k$ is the input of the network. Since the differentiable FoV renders the posterior information matrix at next time step dependent on the prior mean and information matrix of the target state, the input of the network is designed to include them. Finally, we consider the following problem for policy optimization.

Problem Given a prior Gaussian distribution for moving target $\mathbf{y}^{(j)} \sim \mathcal{N}(\mathbf{p}_0^{(j)}, (P_0^{(j)})^{-1})$ with mean $\boldsymbol{\mu}_0^{(j)} \in \mathbb{R}^{n_y}$ and information matrix $Y_0^{(j)} \in \mathbb{S}_{\succ 0}^{n_y \times n_y}$ for all $j \in \{1, \dots, n_l\}$, optimize the parameters $\boldsymbol{\theta} \in \mathbb{R}^{n_p}$ of a control policy $\mathbf{u}_k = \boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{s}_k)$ where $\mathbf{s}_k = [\log(T_k)^\vee, \{\mathbf{p}_{k+1}^{(j)}, \mathrm{vech}(P_{k+1}^{(j)})\}_{j=1}^{n_l}]$ by solve the following policy optimization problem:

$$\max_{\boldsymbol{\theta} \in \mathbb{R}^{n_p}} \sum_{j=1}^{n_l} \log\det(Y_K^{(j)}), \tag{15}$$

subject to

$$T_{k+1} = T_k \exp(\tau \boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{s}_k)) \tag{16}$$

$$\mathbf{p}_{k+1}^{(j)} = A\mathbf{p}_k^{(j)} + B\boldsymbol{\xi}_k^{(j)}, \tag{17}$$

$$P_{k+1}^{(j)} = (A(Y_k^{(j)})^{-1}A^\top + W_k)^{-1}, \tag{18}$$

$$Y_{k+1}^{(j)} = P_{k+1}^{(j)} + M(T_{k+1}, \mathbf{p}_{k+1}^{(j)}), \tag{19}$$

$$M(T, \mathbf{p}^{(j)}) = \Big(1 - \Phi(d(\mathbf{q}(T, \mathbf{p}^{(j)}), \mathcal{F}))\Big) H^\top V^{-1} H, \tag{20}$$

for all $j \in \{1, \dots, n_l\}$ and $k \in \{0, \dots, K-1\}$, where (20) is derived in Koga et al. (2021), Φ is a probit function (Bishop, 2006), defined by the Gaussian CDF $\Phi : \mathbb{R} \to [0, 1]$, $\Phi(x) = \frac{1}{2}\left[1 + \mathrm{erf}\left(\frac{x}{\sqrt{2}\kappa} - 2\right)\right]$, and d is a signed distance function associated with the FoV $\mathcal{F}$ defined below.

Definition 1 *The* signed distance function $d : \mathbb{R}^3 \to \mathbb{R}$ *associated with a set* $\mathcal{F} \subset \mathbb{R}^3$ *is:*

$$d(\mathbf{q}, \mathcal{F}) = \begin{cases} -\min_{\mathbf{q}^* \in \partial\mathcal{F}} ||\mathbf{q} - \mathbf{q}^*||, & \text{if} \quad \mathbf{q} \in \mathcal{F}, \\ \min_{\mathbf{q}^* \in \partial\mathcal{F}} ||\mathbf{q} - \mathbf{q}^*||, & \text{if} \quad \mathbf{q} \notin \mathcal{F}, \end{cases} \tag{21}$$

where $\partial\mathcal{F}$ *is the boundary of* $\mathcal{F}$.

3. Model-Based RL over Continuous $SE(3)$ Trajectory

We approach the problem in the previous section by the following steps. First, we derive the gradient of the cumulative reward with respect to the policy parameters analytically by utilizing the $SE(3)$ pose kinematics and the mean and information update, similarly to iCR (Koga et al., 2021). Then, we design a neural network architecture to handle multiple targets and to enable learning over a varying number of targets.

3.1. Analytical Policy Gradient

The following proposition provides an update rule for the policy function parameters $\boldsymbol{\theta}$ using the gradient of the reward function in (15) with respect to $\boldsymbol{\theta}$.

Proposition 1 *The gradient-ascent update for solving active exploration (15)–(19) with differentiable field of view (20) is given by*

$$\theta^{(i+1)} = \theta^{(i)} + \gamma^{(i)} \frac{\partial r^{\boldsymbol{\pi}_{\boldsymbol{\theta}}}}{\partial \theta^{(i)}}, \tag{22}$$

where $\gamma^{(i)} \in \mathbb{R}_+$ for all $i \in \{1, \dots, n_p\}$ is a step size, and the gradient is given by

$$\frac{\partial r^{\boldsymbol{\pi}_{\boldsymbol{\theta}}}}{\partial \theta^{(i)}} = \sum_{j=1}^{n_l} \operatorname{tr}\left((Y_K^{(j)})^{-1} \Omega_K^{(j,i)} \right), \tag{23}$$

where

$$\Omega_k^{(j,i)} := \frac{\partial Y_k^{(j)}}{\partial \theta^{(i)}} \in \mathbb{R}^{n_y \times n_y}, \qquad \Lambda_k^{(i)} := \frac{\partial T_k}{\partial \theta^{(i)}} \in \mathbb{R}^{4\times 4} \tag{24}$$

are obtained via:

$$\Lambda_0^{(i)} = 0, \quad \Omega_0^{(j,i)} = 0, \quad \forall i \in \{1, \dots, n_p\}, \quad \forall j \in \{1, \dots, n_l\} \tag{25}$$

$$\begin{aligned} \Omega_{k+1}^{(j,i)} &= (A^\top + Y_k^{(j)} A^{-1} W_k)^{-1} \Omega_k^{(j,i)} (A + W_k A^{-\top} Y_k^{(j)})^{-1} \\ &\quad + \left(\Phi'(d(\mathbf{q}, \mathcal{F})) \frac{\partial d}{\partial \mathbf{q}} \right) \Big|_{\mathbf{q}=\mathbf{q}(T_{k+1}, \mathbf{p}_{k+1}^{(j)})} Q T_{k+1}^{-1} \Lambda_{k+1}^{(i)} T_{k+1}^{-1} \underline{\mathbf{p}_{k+1}^{(j)}} H^\top V^{-1} H, \end{aligned} \tag{26}$$

$$\Lambda_{k+1}^{(i)} = \Lambda_k^{(i)} \exp(\tau_k \pi_{\boldsymbol{\theta}}(\mathbf{s}_k)) + T_k \sum_{j=1}^{6} \mathbf{e}_{6,j}^\top \frac{\partial \boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{s}_k)}{\partial \boldsymbol{\theta}} \mathbf{e}_{n_p,i} \frac{\partial \exp(\tau_k \hat{\mathbf{u}})}{\partial \mathbf{u}^{(j)}} \Big|_{\mathbf{u}=\boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{s}_k)}, \tag{27}$$

where $\mathbf{e}_{n,m} \in \mathbb{R}^n$ is a n-dimensional unit vector whose m-th element is 1 and all others are 0.

Proof Taking the gradient of the reward (15) directly leads to (23) by defining $\Omega_k^{(j,i)}$ as (24), which is the perturbation of the information matrix with respect to the policy parameter. Then, the update equation (26) is derived by taking the gradient of both sides in (19) and in (18). The same can be performed for the $SE(3)$ pose state to derive (27) from the gradient of (16). Note that the prior mean is not affected by the control policy due to the update equation (17) and, hence, we do not need to define the perturbation of the prior mean. ■

3.2. Network Architecture

In order to generalize training and testing to varying number of targets, we utilize a padding and masking scheme that allows tracking an arbitrary number of targets (up to a defined maximum $n_l^{\max}$) while keeping the network architecture unchanged. We consider a fixed-length input vector with $n_l^{\max} \times n_y$ elements for both the target state and target information, where only the first $n_l \times n_y$ elements contain non-zero values. Additionally, a binary *Mask* vector contains instruction

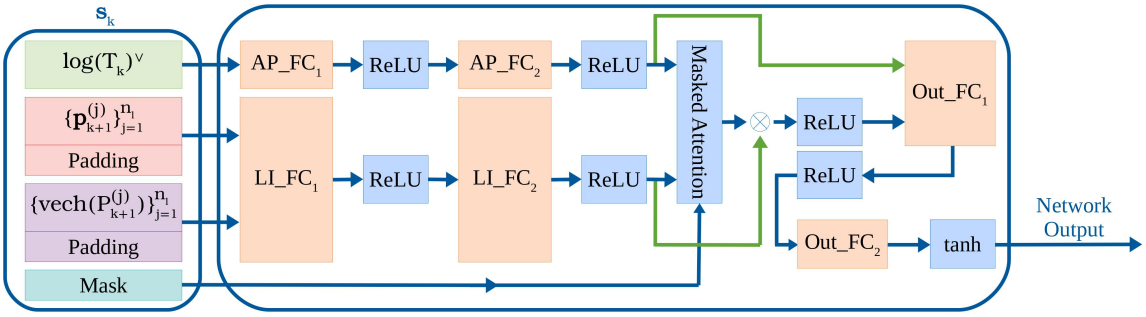

Figure 1: Deep neural network architecture used for the parameterized policy $\boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{s}_k)$. The input $\mathbf{s}_k$ contains the current agent pose in log representation $\log(T_k)^{\vee}$ as well as zero-padded predicted target states $\{\mathbf{p}_{k+1}^{(j)}\}_{j=1}^{n_l}$ and target information $\{P_{k+1}^{(j)}\}_{j=1}^{n_l}$. The *Mask* vector indicates which elements of the padded target input contains relevant values, allowing to remove the influence of zero-padding in the final output. For each input, the network computes continuous controls $\mathbf{u}_k = [\mathbf{v}_k^{\top}, \boldsymbol{\omega}_k^{\top}]^{\top}$.

about which elements of the subsequent computations should be ignored to cancel out the effect of padding values in the output of the network. Fig. 1 illustrates the policy network architecture using the padding and masking scheme, where similarly to Yang et al. (2023) we employ an attention mechanism (Long et al., 2020) so that the agent takes into account the relationship between its current pose state and the moving target states in order to prioritize observing uncertain targets. The fully-connected layers *AP_FC* and *LI_FC* alongside the *ReLU* nonlinearities compute embeddings for the agent pose and target information, denoted as Emb_a and Emb_l respectively. The *Masked Attention* block blends information from the agent, targets, and the masking as follows:

$$\textit{Masked Attention}(\text{Emb}_a, \text{Emb}_l, \text{Mask}) = \textit{softmax}\left((1 + \log(\text{Mask})) \odot \left(\frac{\text{Emb}_a \text{Emb}_b^{\top}}{\alpha}\right)\right), \quad (28)$$

where the operator $\odot$ denotes to element-wise vector multiplication and α is a network hyperparameter. As (28) shows, the components affected by padding can be nullified via the element-wise multiplication since the softmax operator eliminates the influence of the components corresponding to the zero elements of *Mask* in the subsequent matrix multiplication with Emb_l. Therefore, for any input vector $\mathbf{s}_k$ with an arbitrary choice of targets, the policy network $\boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{s}_k)$ computes continuous controls $\mathbf{u}_k = [\mathbf{v}_k^{\top}, \boldsymbol{\omega}_k^{\top}]^{\top}$.

4. Experiments

In this section, we examine the performance of both our proposed model-based RL and a benchmark model-free RL for active target tracking. We provide simulation results to visualize the tracking trajectories and quantitative comparison results of the reward value to demonstrate the robust performance of our model-based RL method.

4.1. Experiment Settings

In the evaluation, we consider 2-D target tracking using a ground vehicle governed by $SE(2)$ differential-drive kinematics and 2-D target positions as the target states ($n_y = 2$). As in practice, we control only the agent's forward and angular speeds within a limited range for linear velocity $v_x \in [0, 4]$ m/s and angular velocity $\omega \in [-\pi/3, \pi/3]$ rad/s. The agent is equipped with a sensor detecting the relative position from the agent to the targets within a triangular FoV with depth of 2 m and angular range of $2\pi/3$ rad/s. When the targets are inside the agent's FoV, their estimated position is updated based on (12).

We set the matrices in the target model as $A = I$ and $B = I$. Regarding the known input $\boldsymbol{\xi} \in \mathbb{R}^2$, we tested two cases:

1) **Unbiased motion:** The known input $\boldsymbol{\xi}$ is sampled from a uniform distribution as $\boldsymbol{\xi} \sim \text{Uniform}[-\bar{\xi}, \bar{\xi}]$, i.e., the mean velocities are 0 and hence the target motion is unbiased. The targets move within small areas based on their current position at every episode.

2) **Biased motion:** The absolute mean of the uniform distribution is set to larger than 0, i.e., the targets have a base linear velocity heading to the same direction with small randomness at each time step. With this setting, the targets do not move too far from other targets during tracking.

We trained the neural network policy only from the biased target motion, because the mean of the uniform distribution is also sampled from the uniform distribution with zero mean, which includes the case of unbiased motion. Besides, the same hyper-parameters, environment settings, and network architecture were applied for model-free and model-based training. Specifically, regarding the environment, the smoothing factor κ, the magnitude of the Gaussian sensor noise σ_1, and the magnitude of the Gaussian motion noise σ_2 were set with values of 0.4, 0.2, and 0.05, respectively. Besides, the time horizon and the targets' initialization position boundaries were set to the same constants dependent on the number of targets which is varied between $[3, 8]$ at each episode during training. For the policy network, the fully-connected layers *AP_FC*$_1$, *AP_FC*$_2$, *LI_FC*$_1$, *LI_FC*$_2$, *Out_FC*$_1$, and *Out_FC*$_2$ have 32, 32, 64, 32, 64 and 2 units, respectively. We choose a hyper-parameter $\alpha = 4$ throughout our experiments. For the model-free reinforcement learning baseline, we apply PPO (Schulman et al., 2017) for training, while the model-based policy was directly trained using gradient ascent over a batch of the last 20 episodes of an epoch, without a replay buffer.

Table 1: Comparison of the proposed model-based RL with the model-free RL. The table shows the average and standard derivation for normalized rewards.

Method	Target Motion Model	3 Targets	5 Targets	7 Targets
		Episodic Reward	Episodic Reward	Episodic Reward
Model-free RL	Unbiased Motion	4.57 ± 2.13	3.65 ± 1.54	1.94 ± 1.70
	Biased Motion	4.23 ± 1.77	3.27 ± 1.30	2.01 ± 1.67
Model-based RL	Unbiased Motion	$\mathbf{6.71 \pm 1.47}$	$\mathbf{6.56 \pm 0.55}$	$\mathbf{5.41 \pm 0.85}$
	Biased Motion	$\mathbf{6.87 \pm 1.21}$	$\mathbf{5.96 \pm 0.96}$	$\mathbf{4.92 \pm 1.07}$

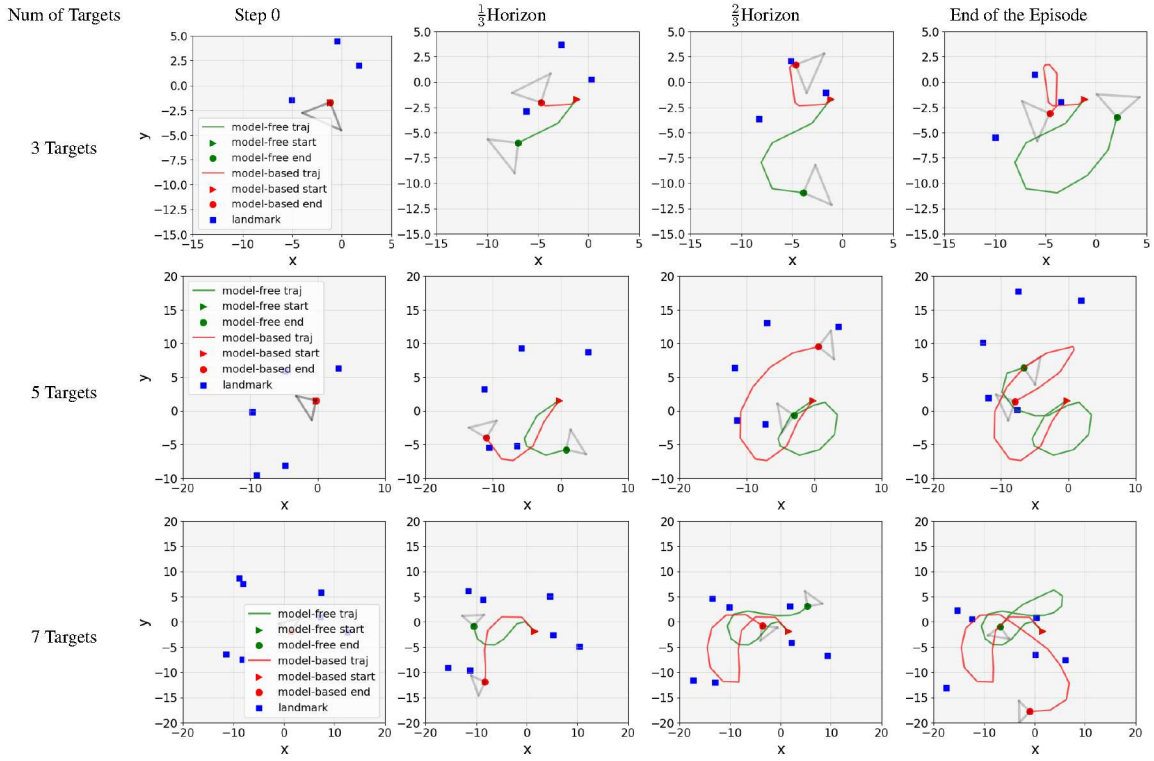

Figure 2: Sensing agent trajectories for target tracking. Two methods are compared in three scenarios with 3, 5, and 7 targets. The blue squares represent the moving targets. The green and red curves show the sensing trajectories generated by model-free and model-based trained networks. The grey triangles illustrate the agents' forward triangular field of view.

4.2. Comparison Results and Analysis

As shown in Fig. 2, we compare the trajectories generated by model-free and model-based trained networks in three scenarios of 3, 5, and 7 targets. In each scenario, the targets' initial position and velocity and the agent's initial pose are identical for fair comparisons. We can clearly see that the network trained with the model-based algorithm is capable of controlling the agent in a better manner for target tracking, while the network trained with the model-free algorithm renders the agent prone to move in a small area.

Quantitative comparisons are shown in Table 1. The metric of the episodic reward is computed based on (15). We normalized the value by dividing by the number of targets at the end of each episode. We chose three random seeds 0, 10, and 100 for both algorithms and tested all the models with two target motions in each scenario. With each setting in one scenario, both methods were tested for 30 runs. According to Table 1, model-based RL has an overall better performance in terms of the larger episodic reward with smaller variance for both unbiased and biased target motion. It is also observed that the average reward is inversely proportional to the number of targets for both methods. We conjecture that when the number of targets increases while the size of the map is also larger, efficient planning for target tracking becomes more challenging accordingly.

5. Conclusion

This paper proposed a model-based reinforcement learning algorithm for tracking multiple dynamic targets using a mobile agent with limited FoV. The prior and posterior mean and information matrix of each target state were obtained by Kalman filtering. We derived an analytical gradient of the target entropy cost function with respect to the parameters of the control policy network by introducing a differentiable FoV and using perturbation of the $SE(3)$ state and the information matrix to obtain a continuous control policy. We observed that our model-based RL algorithm achieves better multi-target tracking in a simulated environment than a model-free RL algorithm based on proximal policy optimization. In future research, we will consider learning the policy for an unknown number of targets, in the presence of obstacles in the environment, and for target tracking by a team of agents with limited communication.

Acknowledgments

We gratefully acknowledge support from NSF FRR CAREER 2045945 and ARL DCIST CRA W911NF17-2-0181.

References

Arash Asgharivaskasi, Shumon Koga, and Nikolay Atanasov. Active mapping via gradient ascent optimization of shannon mutual information over continuous $SE(3)$ trajectories. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 12994–13001, 2022.

Nikolay Atanasov, Jerome Le Ny, Kostas Daniilidis, and George J Pappas. Information acquisition with sensing robots: Algorithms and error bounds. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 6447–6454, 2014.

Nikolay Atanasov, Jerome Le Ny, Kostas Daniilidis, and George J Pappas. Decentralized active information acquisition: Theory and application to multi-robot SLAM. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4775–4782, 2015.

Timothy D Barfoot. *State estimation for robotics*. Cambridge University Press, 2017.

Christopher M Bishop. *Pattern recognition and machine learning*. Springer, 2006.

Cesar Cadena, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, José Neira, Ian Reid, and John J Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on Robotics*, 32(6):1309–1332, 2016.

Xiaoyi Cai, Brent Schlotfeldt, Kasra Khosoussi, Nikolay Atanasov, George J Pappas, and Jonathan P How. Non-monotone energy-aware information gathering for heterogeneous robot teams. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 8859–8865, 2021.

Devendra Singh Chaplot, Dhiraj Gandhi, Saurabh Gupta, Abhinav Gupta, and Ruslan Salakhutdinov. Learning to explore using active neural slam. *arXiv preprint arXiv:2004.05155*, 2020.

Fanfei Chen, John D Martin, Yewei Huang, Jinkun Wang, and Brendan Englot. Autonomous exploration under uncertainty via deep reinforcement learning on graphs. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6140–6147, 2020.

Timothy H Chung, Geoffrey A Hollinger, and Volkan Isler. Search and pursuit-evasion in mobile robotics. *Autonomous robots*, 31(4):299–316, 2011.

Ben Grocholsky, James Keller, Vijay Kumar, and George Pappas. Cooperative air and ground surveillance. *IEEE Robotics & Automation Magazine*, 13(3):16–25, 2006.

Christopher D Hsu, Heejin Jeong, George J Pappas, and Pratik Chaudhari. Scalable reinforcement learning policies for multi-agent control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4785–4791, 2021.

Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. *Advances in Neural Information Processing Systems*, 32, 2019.

Heejin Jeong, Brent Schlotfeldt, Hamed Hassani, Manfred Morari, Daniel D Lee, and George J Pappas. Learning q-network for active information acquisition. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6822–6827, 2019.

Kyle D Julian and Mykel J Kochenderfer. Distributed wildfire surveillance with autonomous aircraft using deep reinforcement learning. *Journal of Guidance, Control, and Dynamics*, 42(8):1768–1778, 2019.

Yiannis Kantaros, Brent Schlotfeldt, Nikolay Atanasov, and George J Pappas. Asymptotically optimal planning for non-myopic multi-robot information gathering. In *Robotics: Science and Systems (RSS)*, pages 22–26, 2019.

Shumon Koga, Arash Asgharivaskasi, and Nikolay Atanasov. Active exploration and mapping via iterative covariance regulation over continuous $SE(3)$ trajectories. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2735–2741, 2021.

Shumon Koga, Arash Asgharivaskasi, and Nikolay Atanasov. Active SLAM over continuous trajectory and control: A covariance-feedback approach. In *American Control Conference (ACC)*, pages 5062–5068, 2022.

Vijay Kumar, Daniela Rus, and Sanjiv Singh. Robot and sensor networks for first responders. *IEEE Pervasive computing*, 3(4):24–33, 2004.

Jerome Le Ny and George J Pappas. On trajectory optimization for active sensing in gaussian process models. In *IEEE Conference on Decision and Control (CDC)*, pages 6286–6292, 2009.

Sergey Levine and Pieter Abbeel. Learning neural network policies with guided policy search under unknown dynamics. *Advances in neural information processing systems*, 27, 2014.

Sergey Levine and Vladlen Koltun. Guided policy search. In *International Conference on Machine Learning (ICML)*, pages 1–9. PMLR, 2013.

Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 17(1):1334–1373, 2016.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Max Lodel, Bruno Brito, Alvaro Serra-Gómez, Laura Ferranti, Robert Babuška, and Javier Alonso-Mora. Where to look next: Learning viewpoint recommendations for informative trajectory planning. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4466–4472, 2022.

Qian Long, Zihan Zhou, Abhibav Gupta, Fei Fang, Yi Wu, and Xiaolong Wang. Evolutionary population curriculum for scaling multi-agent reinforcement learning. *arXiv preprint arXiv:2003.10423*, 2020.

Jianlan Luo, Eugen Solowjow, Chengtao Wen, Juan Aparicio Ojea, Alice M Agogino, Aviv Tamar, and Pieter Abbeel. Reinforcement learning on variable impedance controller for high-precision robotic assembly. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3080–3087, 2019.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

Julio A Placed, Jared Strader, Henry Carrillo, Nikolay Atanasov, Vadim Indelman, Luca Carlone, and José A Castellanos. A survey on active simultaneous localization and mapping: State of the art and new frontiers. *IEEE Transactions on Robotics*, 2023.

Brent Schlotfeldt, Dinesh Thakur, Nikolay Atanasov, Vijay Kumar, and George J Pappas. Anytime planning for decentralized multirobot active information gathering. *IEEE Robotics and Automation Letters*, 3(2):1025–1032, 2018.

Brent Schlotfeldt, Nikolay Atanasov, and George J Pappas. Maximum information bounds for planning active sensing trajectories. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4913–4920, 2019.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv:1707.06347*, 2017.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

Yujin Tang and David Ha. The sensory neuron as a transformer: Permutation-invariant neural networks for reinforcement learning. *Advances in Neural Information Processing Systems*, 34: 22574–22587, 2021.

Jinkun Wang and Brendan Englot. Autonomous exploration with expectation-maximization. In *Robotics Research*, pages 759–774. Springer, 2020.

Pengzhi Yang, Yuhan Liu, Shumon Koga, Arash Asgharivaskasi, and Nikolay Atanasov. Learning continuous control policies for information-theoretic active perception. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2023.