Particle Filter Guided State Estimation for Multi-Agent Reinforcement Learning.

Tirthankar Mittra
University of Colorado Boulder
timi5773@colorado.edu

Abstract

Many multi-agent reinforcement learning (MARL) tasks are characterized by multiple agents operating based on partial incomplete noisy observations. This paper addresses the challenge of state estimation in MARL environments characterized by partial observations through the utilization of a particle filter-type algorithm. The proposed approach aims to enhance the agents' ability to infer the underlying state of the environment based on incomplete information, thereby contributing to improved decision-making and overall performance in complex multi-agent systems. The effectiveness of the proposed method is evaluated through rewards collected by agents in well-known multi-agent reinforcement learning environments like level-based foraging and multi-robot warehouse tasks.

1 Introduction

The inspiration for this paper stems from the operational principles of real-world robots, which can be broadly dissected into two integral components: state estimation and motion planning. In the context of state estimation, robots are designed to assess the state of their environment and determine their pose within it. Following state estimation, traditional robotic systems typically employ motion planning to facilitate interaction with the environment. However, in the present work, the conventional motion planning paradigm is replaced with reinforcement learning, thereby introducing a novel approach to solving reinforcement learning tasks. By incorporating state estimation, we enhance the robustness and efficacy of reinforcement learning algorithms by outsourcing the process to a proven particle filter-type algorithm. This strategic integration not only bolsters the reliability of the algorithms but also leads to superior outcomes. While the focus of this paper primarily delves into the intricacies of multi-agent reinforcement learning tasks, because of their complexity and dynamism in comparison to single-agent reinforcement learning tasks, it is noteworthy that our approach should also exhibit comparable effectiveness in addressing single-agent scenarios.

Reinforcement learning serves as the cornerstone of this study, with the overarching objective of discovering an optimal policy that guides the agent's interactions with the environment, ultimately maximizing the cumulative reward. Leveraging the state estimation aspect of robotic functionality, this paper explores the integration of reinforcement learning into the decision-making process, opening new avenues for adaptive and intelligent robotic behavior.

To evaluate the efficacy of the proposed algorithm, two well-established multi-agent reinforcement learning tasks are employed as test beds. The first task involves a level-based foraging scenario, while the second task simulates a complex multi-robot warehouse environment. Through rigorous experimentation and comparison with established methodologies, this paper seeks to contribute valuable insights into the potential of reinforcement learning in enhancing robotic adaptability and performance across diverse scenarios.

1.1 Level Based Foraging

This task involves multiple agents navigating a grid-world environment with the presence of various food items. The primary objective for the agents is to efficiently collect as much food as possible within the shortest time frame. Notably, each agent and food item in the grid possesses an associated level, introducing a hierarchical dynamic to the task. Specifically, an agent can only collect a food item if its level is less than or equal to the level of the agent.

Collaboration among agents is encouraged, as they may need to combine their efforts to collect food items with levels exceeding that of any individual agent. In instances of successful cooperation, each contributing agent is rewarded proportionally to its contribution, calculated as the product of the total reward and the agent's specific contribution to the collaborative effort. This unique environment encapsulates a delicate balance between cooperation and competition, requiring each agent to optimize its reward while simultaneously fostering collaboration with others to achieve collective success.

Adding to the complexity of the task, each agent is equipped with only partial observations of its immediate surroundings within the grid. To further challenge the decision-making process, we have modified the existing environment to introduce noise to the partial observations made by each agent, compelling them to adapt strategies effectively despite imperfect information. The agents have six available actions: 'do nothing,' 'move north,' 'move south,' 'move east,' 'move west,' and 'load food.' Figure [1] illustrates the environment of the level-based foraging task, depicting two agents within the diagram, each characterized by distinct levels of one and two. The depicted agents are observed navigating toward the direction of a food item of level two.

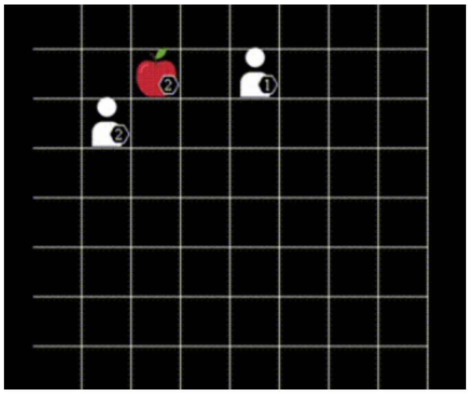

Figure 1: Picture of level-based foraging environment.

1.2 Multi Robot Warehouse Task

2 Methodology

Within the framework of a multi-agent reinforcement learning scenario involving **n** agents, the sequence of interactions unfolds as follows: At time **t-1**, each agent independently selects an action guided by its policy. Subsequently, the environment reacts by providing a set of partial observations and rewards tailored to each agent's actions. The collective action at time **t-1** (u_{t-1}), the joint partial observations at time **t** (z_t), and the joint reward at time **t** (r_t) are expressed through the following equations.

$$
\begin{aligned}
u_{t-1} &= \{a^0_{t-1}, a^1_{t-1}, ..., a^n_{t-1}\} \\
z_t &= \{o^0_t, o^1_t, ..., o^n_t\} \\
r_t &= \{r^0_t, r^1_t, ..., r^n_t\}
\end{aligned}
$$

The task of a particle filter-based state estimation algorithm is to take the set of possible previous state estimates X_{t-1}, the joint actions in that previous state u_{t-1}, and the current set of partial observations z_t to predict current state estimates X_t, this is formalized by Eq[1].

$$X_t = particle\_filter(X_{t-1}, u_t, z_t) \tag{1}$$

In the particle filter-based algorithm letś consider there are **J** particles. Then each particle in set $X_t = \{(x^0_t, wt^0_t), (x^1_t, wt^1_t), ..., (x^J_t, wt^J_t)\}$ at time **t** is an estimation of the actual state x^a_t with probability of estimate $\{wt^0_t, wt^1_t, ..., wt^J_t\}$. The pseudocode of our algorithm is shown below.

Algorithm 1 particle_filter

```
1:  function PARTICLE_FILTER($X_{t-1}, u_{t-1}, z_t$)
2:      $X_t, lgt_t = \{\phi\}, \{\phi\}$
3:      for $i = 1$ to $J$ do
4:          $x^i_{t-1}, wt^i_{t-1} = X_{t-1}[i]$
5:          $x^i_t = motion\_model(x^i_{t-1}, u_{t-1})$
6:          $lgt_t[i] = \sum_{j=0}^{n} \frac{1}{\sqrt{1+(inv\_model(x^{ij}_t)-o^j_t)^2}}$
7:          $X_{t-1}[i] = (x^i_{t-1}, wt^i_{t-1})$
8:      end for
9:      for $i = 1$ to $J$ do
10:         $wt^i_t = \alpha * wt^i_t + (1-\alpha) * \frac{e^{lgt_t[i]}}{\sum_{j=0}^{J} e^{lgt_t[j]}}$
11:         $X_{t-1}[i] = (x^i_{t-1}, wt^i_{t-1})$
12:         $min\_wt = min(min\_wt, wt^i_t)$
13:     end for
14:     $X_{t-1} = sort(X_{t-1}, key = wt_{t-1})$
15:     $X_t = X_t \cup X_{t-1}[0 : \beta * J]$
16:     $X_t = X_t \cup (mutate(X_{t-1}[0 : (1-\beta) * J]), min\_wt)$
17:     return $X_t$
18: end function
```

The *motion_model* function in the above algorithm changes the previous state x^i_{t-1} to the current state x^i_t by using the joint action u^{t-1} and the physics of the reinforcement learning environment. Whereas the *inv_model* function converts each state x^i_t to n partial observations $\{o^{0'}_t, o^{1'}_t, ...o^{n'}_t\}$, where n is the number of agents. Based on the partial observations z_t at time **t**, the logits lgt_t are updated using equation Eq[2].

$$lgt_t[i] = \sum_{j=0}^{n} \frac{1}{\sqrt{1 + (inv\_model(x^{ij}_t) - o^j_t)^2}} \tag{2}$$

Then using these updated logits the probabilities are updated as shown in Eq[3].

$$wt^i_t = \alpha * wt^i_t + (1 - \alpha) * \frac{e^{lgt_t[i]}}{\sum_{j=0}^{J} e^{lgt_t[j]}} \tag{3}$$

In Eq[3] α is used to weigh the historical probability of the previous stateś value. Now each particle is sorted based on their updated weights $X_{t-1} = sort(X_{t-1}, key = wt_{t-1})$, $\beta * J$ of these particles are chosen for the new set X_t, $X_t = X_t \cup X_{t-1}[0 : \beta * J]$. $(1 - \beta) * J$ remaining particles are selected based on mutating the $(1 - \beta) * J$ best particles and assigning min_wt to each of these new particles. The parameter β is used to control exploration versus exploitation of state estimations. J, α, β these are all hyperparameters that were chosen based on trial and error on what works best.