Multi-robot Information Sampling Using Deep Mean Field Reinforcement Learning

Tuffa Said, Jeffery Wolbert, Siavash Khodadadeh,
Ayan Dutta, O. Patrick Kreidl, Ladislau Bölöni, Swapnoneel Roy

***Abstract*— We study the problem of information sampling of an ambient phenomenon using a group of mobile robots. Autonomous robots are being deployed for various applications such as precision agriculture, search-and-rescue, among others. These robots are usually equipped with sensors and tasked with collecting maximal information for further data processing and decision making. The studied problem is proved to be NP-Hard in the literature. To solve the stated problem approximately, we employ a multi-agent deep reinforcement learning framework and use the concepts of mean field games to potentially scale the solution to larger multi-robot systems. Simulation results show that our presented technique easily scales to 10 robots in a 19×19 grid environment, while consistently sampling useful information.**

I. Introduction

Exploring an unknown environment for sampling information about an ambient phenomenon is a fundamental task of a multi-robot system. This has numerous potential real-world applications including search-and-rescue, monitoring, precision agriculture, among others. Robots equipped with sensors can be deployed to collect useful information (e.g., images, videos) and communicate back to a base station for future decision making [3]. However, such information collection using a multi-robot system in an optimal fashion is proved to be an NP-Hard problem [6]. On the other hand, if multiple mobile robots are deployed, they need to effectively coordinate among each other to collect non-redundant information. Researchers in this domain have mostly modeled the underlying information via Gaussian Processes (GPs) and posed the maximal information collection objective as an entropy maximization problem [3], [9], [12], [14], [16].

Learning the model of a singular ambient phenomenon using multiple entities poses its own set of challenges, including joint action learning for intelligent coordination among the robots while ensuring non-redundant path planning. To this end, we propose a novel multi-agent reinforcement learning framework where each robot learns its best local actions using a deep recurrent neural network [4] and an intelligent coordination strategy among the agents is developed using Mean Field games [19], a game theoretical concept. Joint multi-agent reinforcement learning, similar to our problem setting, is shown to lack scalability as it aims to learn the best joint-action of all the involved agents [2]. Combined with the fact that multi-robot information collection itself is computationally intractable, most existing solutions either do not scale well with the number robots or employ greedy-like techniques that may not realize enough value of cooperation.

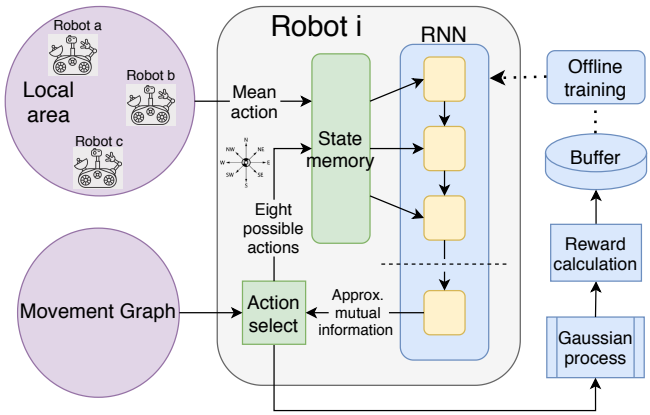

Fig. 1. Overview of proposed MFG-RL system for information collection.

The key idea in a Mean Field game (MFG) is to forego learning as a function of all other robots' actions, learning rather as a function of the mean action of all other robots. In effect, the n-agent learning problem reduces to a 2-robot learning scenario, making it highly scalable, but the exact details of this reduction can impact whether the resulting strategy remains effective. Our proposed solution uses the MFG ideas along with a Recurrent Neural Network (RNN) for effective multi-robot information sampling. An overview of the proposed system is shown in Fig. 1. Results show that the presented technique consistently outperforms other existing sampling solutions in terms of collected information, while not requiring the robots to run compute-intensive prediction algorithms on-board during deployment.

This paper's contributions are summarized as follows:

- The first work to use the ideas of multi-agent learning and mean field reinforcement learning for multi-robot informative path planning.
- The first work to integrate recurrent neural networks with mean-field reinforcement learning, although convolutional neural neural networks and mean field learning have been recently considered [19].
- Numerical results that show our proposed strategy easily scales up to 10 robots, while yielding higher reward than

T. Said, J. Wolbert, A. Dutta, O.P. Kreidl, and S. Roy are with the University of North Florida, USA. Emails: {n01042011, n01424561, a.dutta, patrick.kreidl, s.roy}@unf.edu
S. Khodadadeh and L. Bölöni are with the University of Central Florida, USA. Emails: siavash.khodadadeh@knights.ucf.edu, ladislau.boloni@ucf.edu
This work was supported by the National Science Foundation (NSF) CPS Grant #1932300.

comparable techniques.

II. Related Work

Although multi-robot path planning has been studied for decades, only recently have researchers looked into a more applied variant—multi-robot informative sampling. The earliest work is due to [11], which proposed a branch-and-bound solution for multi-robot informative path planning under limited budget constraints. It is an offline approach, meaning the robots commit to the paths before making any actual observations. This is possible due to the non-adaptive nature of their optimization metric. The authors used a Gaussian Process to model the underlying phenomenon. Online, adaptive and decentralized solutions have also been proposed: in [14], the authors have accounted for noisy sensors and the robots sample the same location multiple times to learn the noise model, while in [15] the authors have used a wind field model to test the proposed approach. Dutta et al. have proposed a solution for online informative path planning with a continuous connectivity constraint i.e., the robots have to be within each other's communication ranges throughout exploration [3]. A similar study appears in [5] except that robots are constrained to regain connectivity only periodically, otherwise free to explore individually.

Recently, Wei and Zheng [17] have presented a reinforcement learning technique for informative sampling with a single robot. Our paper extends this reinforcement learning idea to n robots, also using the game-theoretic concepts of mean field games (MFGs) [7] to permit larger n. Mayya et al. used mean field approximations to model inter-robot collisions as sources of information [10]. Our paper uses mean field approximations to model multi-agent reinforcement learning, inspired by the approach taken in [19] albeit their reward function does not express informative sampling objectives. Finally, Li et al. [8] used MFG to maximize driver efficiency in ride-share dispatching by a deep-Q actor-critic system. The mean field force is called the "average response" in this case and is the ratio of agents/orders in a locale centered on the agent. The mean field was not used in online planning, but was used in the training of the centralized critic network.

III. Problem Setup

We have a set of n mobile robots $R = \{r_1, \cdots, r_n\}$ exploring a polygonal environment E to collect information. As detailed further in Subsection III-A, the environmental information is modeled over a finite set of discrete Points Of Interest (POIs). The POIs (as vertices) induces a physical movement graph $G_p = \{\text{POIs}, e, w\}$, where e represents the edges (logical routes among the POIs) and w represents the edge weights (travel distances among the POIs). Specifically, from any POI v, graph G_p implies the available action set $A(v)$ of movement directions by which the robot traverses to a neighboring POI—if POIs lie on a grid, for example, then action set $A(v)$ specializes to the cardinal directions. The robots start exploration from unique locations $S = \{s_1, \cdots, s_n\} \subset$ POIs, each ending its exploration upon expiring a given budget B (e.g., bound on total travel distance). Let P_i denote the path (i.e., ordered set of vertices in POIs starting from s_i) followed by robot r_i such that $cost(P_i) \leq B$, where $cost$ is calculated as the sum of the edge weights in G_p along path P_i. As detailed further in Subsection III-B, during exploration the robots make movement choices sequentially with time. The associated opportunities for a robot r_i to coordinate subsequent exploration with other robots $R_{-i} = R \backslash r_i$ will be featured in Section IV.

A. Environmental Information as a Gaussian Process

The environment's information is modeled using a Gaussian Process (GP), which associates to the collection locations (POIs) a Gaussian random vector $\mathbf{Z}$ having covariance matrix Σ (and zero mean). Its (differential) entropy, a volumetric measure of the prior uncertainty, is given by

$$H(\mathbf{Z}) = \frac{1}{2}\log|\Sigma| + \frac{|\text{POIs}|}{2}(1 + \log(2\pi)) \quad ,$$

where $|\text{POIs}|$ denotes set cardinality but $|\Sigma|$ denotes matrix determinant. We assume each robot's sensing process (e.g., camera) exhibits negligible noise, so that merely moving to any particular POI resolves all uncertainty in that location. It follows that the POIs can always be partitioned into subsets U and V, corresponding to locations that are unvisited and visited, respectively. The Gaussian random vector $\mathbf{Z}_{U|V}$, characterizing all uncollected information upon visiting locations V, has (posterior) covariance matrix

$$\Sigma_{U|V} = \Sigma_{UU} - \Sigma_{UV}\Sigma_{VV}^{-1}\Sigma_{VU}$$

in terms of prior statistics organized into the corresponding block form with respect to U and V i.e.,

$$\Sigma = \begin{bmatrix} \Sigma_{UU} & \Sigma_{UV} \\ \Sigma_{VU} & \Sigma_{VV} \end{bmatrix}.$$

The implied posterior entropy is

$$H(\mathbf{Z}_{U|V}) = \frac{1}{2}\log|\Sigma_{U|V}| + \frac{|U|}{2}(1 + \log(2\pi))$$

and, in turn, the reduction of uncertainty in the uncollected information $\mathbf{Z}_U$ due to the collected information $\mathbf{Z}_V$ is measured by their mutual information

$$I(\mathbf{Z}_U; \mathbf{Z}_V) = H(\mathbf{Z}_U) - H(\mathbf{Z}_{U|V}) = \frac{1}{2}\log\left(\frac{|\Sigma_{UU}|}{|\Sigma_{U|V}|}\right) \quad (1)$$

It should be noted that the above information model captures two alternative objectives that are common in the informative sampling literature. The objective to decrease overall uncertainty involves choosing visited points V such that the posterior entropy $H(\mathbf{Z}_{U|V})$ is maximized. On the other hand, the objective to inform predictions of uncollected information is achieved by maximizing mutual information $I(\mathbf{Z}_U; \mathbf{Z}_V)$, which also involves leaving unvisited points U that are highly uncertain to begin with (as quantified by $H(\mathbf{Z}_U)$, the marginal prior entropy). However, the above model does *not* explicitly capture the common objective of optimal prediction in the minimum root-mean-square-error sense, which is given by the posterior mean and depends upon the observations realized online. This is because the

informative sampling problem formulated in Section IV is an *offline* learning approach, optimizing with respect to the GP's ensemble of realizations, not just one specific realization.

A final remark concerns the determinant being an expensive computation, scaling roughly cubically with matrix dimensions. It is common (e.g., in kernel-based parametrizations of the GP) that there are dimishing correlations among POIs as the distance between them grows. This induces a diagnonally-dominant covariance matrix and, in turn, motivates approximation of its determinant by the product of the per-POI variances $\sigma_1^2, \sigma_2^2, \sigma_3^2, \ldots$ along its diagonal e.g.,

$$\log|\Sigma_{UU}| \approx \sum_{u=1}^{|U|} \log(\sigma_u^2) \quad .$$

The approximation is, in fact, an upper bound on the true determinant (via Hadamard's inequality), achieving equality if and only if the GP's covariance matrix is truly diagonal.

B. Informative Sampling as a Dynamic Program

Given the physical movement graph G_p and starting locations S of the robots, denote by $\mathcal{P}_i(s_i; B)$ for robot r_i the set of all paths in G_p that are admissible (i.e., start from location s_i and respect the budget $cost(P_i) \leq B$). In turn, denote by $\mathcal{P}(S; B) = \mathcal{P}_1(s_1; B) \times \cdots \times \mathcal{P}_n(s_n; B)$ the set of all admissible multi-robot plans, taking for granted online collision avoidance (and inter-robot coordination) when multiple robots are simultaneously commanded to the same location. Observe that any path P_i in $\mathcal{P}_i(s_i; B)$ implies a subset of visited locations $V(P_i) \subset$ POIs. In turn, any plan P in $\mathcal{P}(S; B)$ implies the corresponding subset $V(P) = \cup_{i=1}^{n} V(P_i)$, leaving $U(P) =$ POIs$\backslash V(P)$ as the subset of unvisited locations. Now, also appealing specifically to Eq. 1, our multi-robot informative sampling problem is stated as

$$P^* = \arg\max_{P \in \mathcal{P}(S;B)} I(\mathbf{Z}_{U(P)}; \mathbf{Z}_{V(P)}) \tag{2}$$

The problem cannot be solved as stated, of course, due to the enormous cardinality of the plan set $\mathcal{P}(S; B)$, especially as the number of robots n gets large. The approximate solution we describe in Section IV starts by casting Eq. 2 into its sequential counterpart, invoking the theory of discrete-time dynamic programming. Specifically, let state $\mathbf{x}(t)$ denote the plan as followed by the robots up until timestep t, the associated set of visited cells denoted by $V(\mathbf{x}(t))$, and let $\mathbf{a}(t)$ denote the joint action the robots next take. Observe that the component state $x_i(t)$ implies robot r_i's current location, call it v_i, from which the movement graph G_p implies its admissible actions $A(v_i)$ and, in turn, upon taking any such action $a_i \in A(v_i)$ the subsequent location $V(a_i)$. The set of admissible joint actions is then expressed by $A(\mathbf{x}(t)) = A(v_1) \times \cdots \times A(v_n)$, where for each such joint action $\mathbf{a} = (a_1, a_2, \ldots, a_n)$ the associated subsequent locations are denoted by $V(\mathbf{a}) = \cup_{i=1}^{n} V(a_i)$. It follows that $V(\mathbf{x}(t+1)) = V(\mathbf{x}(t)) \cup V(\mathbf{a}(t))$ comprises the visited locations up until timestep $t+1$, leaving unvisited locations $U(\mathbf{x}(t+1)) =$ POIs$\backslash V(\mathbf{x}(t+1))$. Altogether, there then exists a function $Q_t^*(\mathbf{x}(t), \mathbf{a}(t))$ by which

$$\mathbf{a}^*(t) = \arg\max_{\mathbf{a} \in A(\mathbf{x}(t))} Q_t^*(\mathbf{x}(t), \mathbf{a}) \tag{3}$$

because (i) the state evolution constraint is iterative over time, where the per-stage system equation is specifically the form of $\mathbf{x}(t+1) = F(\mathbf{x}(t), \mathbf{a}(t))$, and (ii) the mutual information objective in Eq. 1 decomposes additively over time, where the per-stage reward equation is specifically

$$G(\mathbf{x}(t), \mathbf{a}(t)) = I\left(\mathbf{Z}_{U(\mathbf{x}(t+1))|V(\mathbf{x}(t))}; \mathbf{Z}_{V(\mathbf{a}(t))|V(\mathbf{x}(t))}\right). \tag{4}$$

While the optimal Q-function in Eq. 3 is known to obey a certain recursion (involving per-stage equations F and G), its computation scales exponentially with the number of timesteps and robots, prohibiting its application here.

IV. Informative Sampling via Mean Field Reinforcement Learning

Because our reward function (i.e., mutual information Eq. 4) depends only on the environment's statistics (e.g., the covariance matrix), it can be calculated *offline*, or without making the robots explore the environment in reality [6], [17]. In a single-robot setting, the sequential counterpart to the offline informative sampling problem of Eq. 2 can be expressed as $a_i^*(t) = \arg\max_{a_i \in A(x_i(t))} Q_t^*(x_i(t), a_i)$. However, in a multi-robot scenario, a robot r_i's learning will be affected not only by its own actions a_i but also the actions of the other robots R_{-i}. For example, if the robots are too close, they will miss exploring a major portion of the environment. On the other hand, if they are too far (e.g., exploring along the opposite boundaries), the uncertainty in the middle would be too high. Hence, we pose it as a multi-agent reinforcement learning problem, where each robot learns from its own as well as other robots' actions and seeks a policy that maximizes the joint reward.

A. Mean Field Game Formulation

Modeling the information collection problem as a multi-agent learning scenario still faces the challenge that the joint action set $\mathbf{a}$ grows exponentially with n, the number of robots. In turn, it quickly becomes intractable to learn using the standard Q-function presented in Eq. 3. To this end, we use the concepts of mean field games [13], [19] to reduce the problem to pairwise action learning as follows:

$$Q_t^*(\mathbf{x}(t), \mathbf{a}) \approx \frac{1}{|R_{-i}|} \sum_{r_j \in R_i'} Q^*(\mathbf{x}(t), a_i, a_{-i}), \tag{5}$$

where a_{-i} denotes the mean action taken by the robots $r_j \in R_{-i}$. This pairwise approximation can be used while reducing the complexity of joint-action learning significantly [1], [19], where all other robots' actions are similarly averaged:

$$a_{-i} = \frac{1}{|R_{-i}|} \sum_{r_j \in R_{-i}} a_j \tag{6}$$

Each robot r_i stores a vector of all the past mean actions (a_{-i}) calculated from the actions of other robots in R_{-i}. In other words, this pairwise Q-function helps each robot r_i to

learn the best action based on the virtual mean agent that is abstracted by the mean effect of all the robots in R_{-i}. Then, each robot's local pairwise Q-function, called the mean field Q-function, is updated recursively from the training episodes as in classical Q-learning:

$$Q^i_{t+1}(\mathbf{x}(t+1), a_i, a_{-i}) = (1-\alpha)Q^i_t(\mathbf{x}(t), a_i, a_{-i}) + \alpha(G(\mathbf{x}(t), \mathbf{a}(t)) + \gamma \max_{a'_i, a'_{-i}} Q^i_t(\mathbf{x}(t+1), a'_i, a'_{-i}) \quad (7)$$

where α and γ represent the learning rate and the discount factor, respectively; refer to [19] for more details.

B. Deep Q-Learning Implementation

Due to the recent advancement in the field of deep learning and consequently deep reinforcement learning, the Q-function (Eq. 5) can be approximated using a deep neural network – a function approximator. As the robots' future locations for exploration depend on the visited locations in the environment so far, each robot uses a Recurrent Neural Network (RNN) [4] instead of a traditional Convolutional Neural Network (CNN) such as used in [19]. RNNs have been shown to be useful specifically for sequential data sets, such as the robots' (mean) actions in our setting. We have used a hyperbolic tangent activation function ($tanh$). The following notations are adopted from [4], [18]. Let x^t and y^t denote the input and output at timestep t. Let h^t denote the hidden state of the network at timestep t and it is defined as follows: $tanh(\mathbf{b} + \mathbf{W}h^{t-1} + \mathbf{U}x^t)$. The output then is defined as $y^t = \mathbf{c} + \mathbf{V}h^t$, where $\mathbf{b}$ and $\mathbf{c}$ are the bias vectors and $\mathbf{W}, \mathbf{V}$, and $\mathbf{U}$ denote the weight matrices. Output y^t can be probabilities in a normalized form, for example. In our setting, the output is a probability vector of performing discrete actions. We use Long Short Term Memory (LSTM) networks as our RNN architecture. These networks are mostly useful for remembering long history of data. The main idea is to maintain a cell state—data addition or deletion to which is regularized by a gate. The input x^t to the robots' RNNs is a tuple containing their locations and the mean actions of the neighboring robots, a_{-i}. The output of a robot's RNN is a length-$|A|$ vector, each element holding the probabilities of taking actions in timestep $(t+1)$.

The pseudocode of the process is given in Algorithm 1. In each episode of the exploration process, a robot finds a B-length path. In every timestep of the episode, each robot predicts the sensor measurements in the unvisited POIs U. Based on this prediction, they decide the rewards for all the neighbor POIs, i.e., the POIs that they share edges with in G_p. If the prospective POIs are already visited by any of the robots previously, including the robots' current locations, we assign zeros as rewards; otherwise the corresponding mutual information values are calculated (Eq. 4). The robots use a ϵ-greedy policy to explore the environment. Once every robot takes an action and move to the next POI to collect information, the actions are shared with other robots R_{-i}. Next, the mean action of the robots in R_{-i} is calculated and stored in a tuple to be used as an input to the RNN.

The state transition tuples, i.e., the experiences, are stored in a memory buffer $\mathcal{D} = \{e_1, e_2, \cdots, e_t\}$, namely a priority memory replay, where e is a tuple $(\mathbf{x}(t), \mathbf{a}(t), G(\mathbf{x}(t), \mathbf{a}(t)), \mathbf{x}(t+1))$. Batches from $\mathcal{D}$ are sampled to update the network parameters at the end of every episode. While training the network, a random batch of episodes are selected and the experiences generated from those episodes are used to update the Q^* function with gradient decent-based optimization with mean square error acting as the loss function as follows:

$$\mathcal{L}(\{x^1, x^2, \cdots, x^t\}, \{y^1, y^2, \cdots, y^t\}, \theta) = \mathbb{E}_{(\mathbf{x}(t), \mathbf{a}, G, \mathbf{x}(t+1)) \sim \mathcal{D}}[(y^t - Q^*(\mathbf{x}(t), \mathbf{a}(t)), \theta)^2],$$

where we collectively denote the weights and biases in the network with θ. The episode yielding the highest reward is stored as the final solution.

Algorithm 1: Mean Field Reinforcement Learning for Multi-Robot Information Sampling

```
Input : R ←A set of robots; B ←Budget; D ← ∅;
for each episode do
    Episode experience list e ← ∅;
    Initialize robots' starting locations S;
    while B ≥ 1 do
        for r_i ∈ R do
            v_i ← Current location of r_i;
            Initialize the robot's current reward vector G;
            for a_k ∈ A(v_i) do
                v_l ← potential location of r_i after
                  executing action a_k;
                if v_l has been visited by a robot r_j ∈ R
                  then
                    G ← 0;
                else
                    G ← reward based on Eq. 4;
            Decide action a_i* based on ε-greedy policy;
            Execute a_i*;
            Update the current location to v_i* ← V(a_i*);
            Compute the mean action a_{-i} (Eq. 6);
            Update the state x ← (v_i*, a_{-i});
            Update the experience tuple e;
        B ← B − 1;
    Store e in D;
    Sample batch of previous episode experiences from D;
    Update Q* with loss L and Adam optimizer;
```

V. Experimental study

In this section we describe a series of experiments that study the training cost and performance of our approach compared with baselines. We consider n robots moving on a rectangular grid. At every time step, a robot can move to any of the 8 neighboring cells. The centers of the grid cells are considered the nodes of the G_p. For each *exploration episode* the robots start at a random position and follow their movement policy for the number of steps specified by the energy budget B.

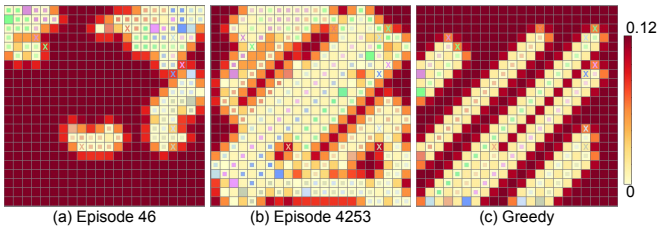

Fig. 2. Three examples of the residual uncertainty the end of an exploration episode. Left: proposed algorithm early in the training, center: proposed algorithm after 4253 training episodes, right: the baseline SSPG algorithm.

The objective of the exploration is to reduce uncertainty about the values of a scalar field mapped to the environment. We represent the uncertainty in the form of a Gaussian process, with observation points (POIs) located at the cells. The uncertainty starts at a maximum value; whenever a cell is visited, the uncertainty, indicated by the variance of the Gaussian process, is reduced to zero. However, the uncertainty also decreases at the nearby points by a factor described by the squared exponential kernel $k_{\text{SE}}(c, c') = \sigma^2 \exp\left(-\frac{|c-c'|^2}{2\ell^2}\right)$. The *reward* obtained by the robot(s) is defined by Eq. 4. Figure 2 shows the residual uncertainty at the end of exploration episodes for different algorithms.

A. Baselines

We compare our proposed method with two baselines:

Random Waypoint (RW): each robot moves towards an independently selected random neighboring waypoint. For this algorithm the robots do not need to communicate and they also don't need information about the Gaussian process and its kernel size.

Shared State Predictive Greedy (SSPG): each robot chooses the next action that maximizes the immediate reward. The robot must have access to the Gaussian process in real time and the Gaussian process must be updated after every action taken by every robot. This requires extensive communication between the robots or between the robots and a centralized controller. Furthermore, SSPG requires the robot to predict the values of the Gaussian function for each possible action it can take.

In contrast to SSPG, our proposed approach does not require real time access to the Gaussian process. The Gaussian process is only used for the calculations of the reward during training and only for the actions actually performed.

B. Experiments

The first set of experiments investigate the scalability of the proposed approach, for which a key contributor is the computational cost of a training episode in a typical scenario with a certain number of robots. Naturally, the training times also vary as a function of the size and shape of the environment and the energy budget of the robots; systems with more robots are usually deployed in larger environments, where the recalculation of the Gaussian Process adds to the training cost. Figure 3 shows the measured per-episode

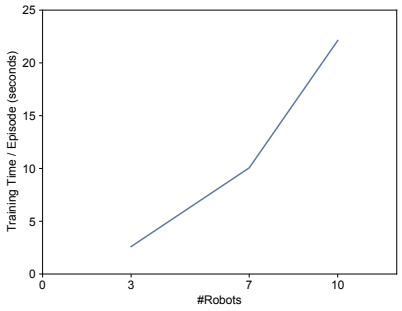

Fig. 3. Per episode training time for typical environment configurations with different number of robots. We find that the training cost is approximately linear with the number of robots.

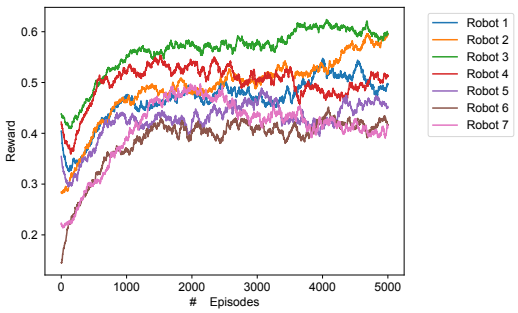

Fig. 4. The convergence of 7 different robots for budget percentage of 40.

training times on a Nvidia Tesla V100 PCIe GPU with 16GB of RAM. We find for typical configurations that the per-episode computational cost increases approximately linearly with the number of robots, a favorable scalability result.

An associated question is the number of episodes necessary for the learning to converge. Figure 4 shows the reward obtained per-robot after a given number of training episodes on a scenario involving 7 robots and a budget of 40% of the maximum. We find that the performance reaches a plateau for all robots at approximately 1500 iterations. The reward collected by the individual robots varies with as much as 50%, due to the initial location of the robots.

Finally, let us compare the performance of our algorithm against the baseline algorithms RW and SSPG. Figure 5 compares the total reward obtained by the three algorithms for two scenarios and a varying number of robots. We notice that, in general, a larger number of robots leads to a larger reward for all algorithms (although occasional variations caused by the random starting locations are possible). We also find that our algorithm outperforms the baselines for the cases with more than three robots. For the case with 3 robots, the SSPG algorithm narrowly outperforms our approach, at the cost of maintaining a Gaussian process at test time.

Finally, we study the distribution of the collected reward among the participating robots. In the scenario shown in Figure 6, we find that for our algorithm there can an as high as $2.5\times$ variation between the reward collected by the different robots. However, this variation is as high as $6\times$ for the SSPG algorithm, due to the myopic nature of greedy trajectories. The random nature of RW, on the other hand,

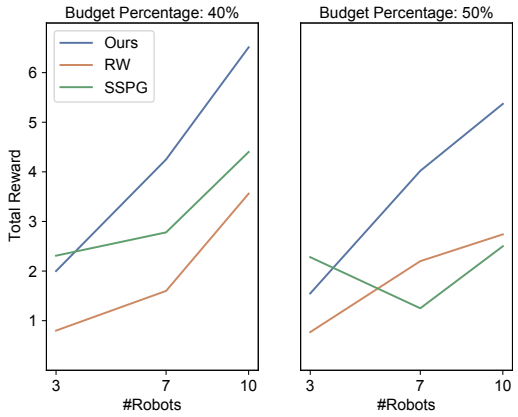

Fig. 5. Comparison of the performance between our algorithm and proposed baselines for different numbers of robots.

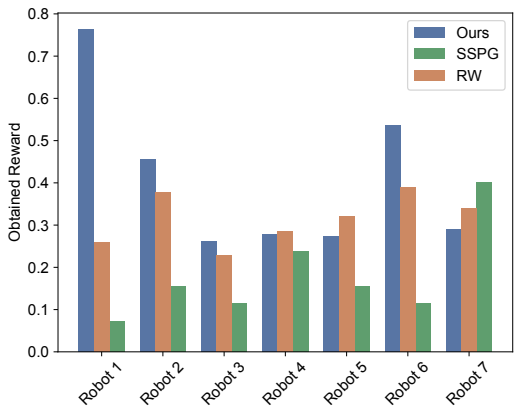

Fig. 6. Reward distribution for seven robots with a budget percentage 50.

creates a more uniform distribution but lower overall reward.

VI. Conclusion and Future Work

We proposed a solution to the multi-robot information sampling problem combining recent works in multi-agent learning and mean field reinforcement learning. Multi-agent learning problems are known to become especially challenging as the number of robots increases: our mean field ideas essentially approximate the n-agent problem by a crafted 2-agent problem. This 2-agent approximation is deliberately formulated to be compatible with contemporary reinforcement learning approaches, but the exact details of the reduction impact whether the resulting strategy remains effective with respect to the original information sampling objectives: our deep Q-learning implementation employs a recurrent neural network with long-short-term-memory architecture, leveraging state-of-the-art deep learning tools during training via offline information collection simulations. The presented experiments study the training cost and sampling quality of our approach, where the results suggest (i) training times scale approximately linearly with the number of robots and (ii) the resulting strategy consistently achieves a higher total reward than a random-waypoint baseline as well as a less-volatile per-robot reward than a shared-state-predictive-greedy baseline. Moreover, these performance comparisons with the two baseline strategies are seen to become more significant as the number of robots increases, consistent with our original motivation to tackle larger-scale information collection scenarios. Future work may consider expanded experimentation, including variable environmental parameters (e.g., size, uncertainty) and training parameters (e.g., rate, discount), as well as alternative mean-field reinforcement learning approximations to the one analyzed here.

References

[1] L. E. Blume et al. The statistical mechanics of strategic interaction. *Games and economic behavior*, 5(3):387–424, 1993.

[2] L. Busoniu, R. Babuska, and B. De Schutter. A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 38(2):156–172, 2008.

[3] A. Dutta, A. Ghosh, and O. P. Kreidl. Multi-robot informative path planning with continuous connectivity constraints. In *Int'l Conf. on Robotics and Automation (ICRA)*, 2019.

[4] M. J. Hausknecht and P. Stone. Deep recurrent Q-learning for partially observable MDPs. In *2015 AAAI Fall Symposia, Arlington, Virginia, USA, November 12-14, 2015*, pages 29–37. AAAI Press, 2015.

[5] G. A. Hollinger and S. Singh. Multirobot coordination with periodic connectivity: Theory and experiments. *IEEE Transactions on Robotics*, 28(4):967–973, 2012.

[6] A. Krause, A. Singh, and C. Guestrin. Near-optimal sensor placements in Gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research*, 9(Feb):235–284, 2008.

[7] J.-M. Lasry and P.-L. Lions. Mean field games. *Japanese journal of mathematics*, 2(1):229–260, 2007.

[8] M. Li, Z. Qin, Y. Jiao, Y. Yang, J. Wang, C. Wang, G. Wu, and J. Ye. Efficient ridesharing order dispatching with mean field multi-agent reinforcement learning. In *The World Wide Web Conference*, 2019.

[9] K.-C. Ma, Z. Ma, L. Liu, and G. S. Sukhatme. Multi-robot informative and adaptive planning for persistent environmental monitoring. In *Distributed Autonomous Robotic Systems*, pages 285–298. Springer, 2018.

[10] S. Mayya, P. Pierpaoli, G. N. Nair, and M. Egerstedt. Collisions as information sources in densely packed multi-robot systems under mean-field approximations. In *Robotics: Science and Systems*, 2017.

[11] A. Singh, A. Krause, C. Guestrin, and W. J. Kaiser. Efficient informative sensing using multiple robots. *Journal of Artificial Intelligence Research*, 34:707–755, 2009.

[12] A. Singh, A. Krause, and W. J. Kaiser. Nonmyopic adaptive informative path planning for multiple robots. In *Twenty-First Int'l Joint Conf. on Artificial Intelligence*, 2009.

[13] J. Subramanian and A. Mahajan. Reinforcement learning in stationary mean-field games. In *Proc. of the 18th Int'l Conf. on Autonomous Agents and MultiAgent Systems (AAMAS)*, 2019.

[14] A. Viseras, T. Wiedemann, C. Manss, L. Magel, J. Mueller, D. Shutin, and L. Merino. Decentralized multi-agent exploration with online-learning of Gaussian processes. In *Int'l Conf. on Robotics and Automation (ICRA)*, pages 4222–4229. IEEE, 2016.

[15] A. Viseras, Z. Xu, and L. Merino. Distributed multi-robot cooperation for information gathering under communication constraints. In *Int'l Conf. on Robotics and Automation (ICRA)*, 2018.

[16] A. Viseras Ruiz, M. Angermann, I. Wieser, M. Frassl, and J. Mueller. Efficient multi-agent exploration with Gaussian processes. In *Australasian Conf. on Robotics and Automation (ACRA)*, 2014.

[17] Y. Wei and R. Zheng. Informative path planning for mobile sensing with reinforcement learning. *arXiv preprint arXiv:2002.07890*, 2020.

[18] J. Xu, R. Rahmatizadeh, L. Bölöni, and D. Turgut. Real-time prediction of taxi demand using recurrent neural networks. *IEEE Trans. on Intelligent Transportation Systems*, 19(8):2572–2581, 2017.

[19] Y. Yang, R. Luo, M. Li, M. Zhou, W. Zhang, and J. Wang. Mean field multi-agent reinforcement learning. In *Proc. of the 35th Int'l Conf. on Machine Learning, ICML*, 2018.