Entropy Maximization in High Dimensional Multiagent State Spaces

Ayhan Alp Aydeniz[1], Enrico Marchesini[2], Robert Loftin[3], and Kagan Tumer[1]

Abstract— **Underwater or planetary exploration are prime examples of missions that can benefit from autonomous agents working together. However, discovering effective team-level behaviors (*i.e.*, coordinated joint actions) is challenging in these domains as agents typically receive a sparse reward (zero–or constant–for the majority of the interactions). To address this issue, intrinsic rewards encourage agents to explore diverse policies to visit the state space more effectively. Unfortunately, as the agents' state space grows, intrinsic reward-based (*i.e.*, curiosity) approaches become less effective as they cannot effectively distinguish a *diverse* set of states. In this direction, we introduce state entropy maximization for multiagent learning where agents explore using local (dense) rewards and learn to solve the coordination task by leveraging global (sparse) rewards. Because of the intrinsic ability to balance local and global rewards, our approach enables the state entropy function to remain effective in high dimensional state spaces. Experiments in tightly coupled tasks requiring complex joint actions, show that local entropy-based rewards enable agents to discover successful team behaviors in high dimensional spaces where previous hand-tuned count-based rewards fail.**

I. Introduction

Multiagent systems (MAS) have the potential to accelerate research in a variety of remote and exotic domains, such as exoplanets [1] and the depths of the ocean [2]. Such potential lies in the autonomous coordination of the agent teams, that allows us to expand the search horizons of our systems. To accomplish these coordination behaviors, we typically design exploration strategies to learn optimal joint actions that can successfully achieve collective objectives (*i.e.*, solving desired tasks) [3], [4]. However, finding such effective joint policies is inherently difficult, especially (i) when scaling up the problem complexity (*e.g.*, the dimension of the state space, the size of the environment, and the number of agents) and (ii) in tightly-coupled MAS where agents have to interact with each other in a highly coordinated manner. Therefore, developing novel exploration methods to discover such policies remains an open challenge.

In this direction, Deep Reinforcement Learning (RL) has achieved remarkable progress in terms of exploration and performance in single-agent settings. A leading approach for promoting exploration in RL is *intrinsic motivation* [5] that considers maximizing a signal based on the state novelty (*e.g.*, the number of visits to a particular state). However, it is challenging to scale intrinsic motivation in high dimensional continuous environments since a visit to a particular state will likely occur once. Recently, *exploration-driven* methods address such an issue by modeling the state space using a variety of techniques (*e.g.*, density models, *k-nearest neighbor* search or count-based rewards) [6], [7], [8], [9], [10]. Nonetheless, despite being effective exploration methods, these solutions cannot be directly applied to multiagent settings due to the size of the problem and the challenge of learning both team and agent-level objectives [11], [12].

We address these issues by proposing State Entropy Maximizing Multiagent Evolutionary Reinforcement Learning (SEM-MERL). In contrast to the common practice of addressing the exploration-exploitation dilemma as a single problem, we abstract it into multiple (two) levels [13]. In more detail, *individual agents* maximize a state entropy signal to efficiently explore the environment and provide diverse sets of experiences to a *team-level* Evolutionary Algorithm (EA). The EA maintains a population of entropy-maximizing multiagent teams (*i.e.*, a population of teams), and its goal is to discover cooperative behaviors that lead to higher payoffs. In particular, the proposed framework achieves scalability by computing a *k-nearest neighbor* estimate of the entropy [14], [15] at the per-agent level. In addition, SEM-MERL maintains a uniform state distribution and makes the proposed method applicable to high dimensional multiagent tasks. In contrast, previous methods such as count-based rewards [13] typically use a discrete distribution of states (or quantization), hindering performance in long-duration and high dimensional tasks. SEM-MERL allows us to apply this hierarchical strategy in high dimensional state spaces.

To show the benefits of SEM-MERL, we conduct numerous experiments in increasingly complex variations of a well-established multi-robot cooperative exploration domain called *multi-rover exploration* [16]. Here, agents are required to take closely coordinated actions to observe different points of interest simultaneously. Crucially, our experiments show that SEM-MERL successfully covers a significantly larger portion of the behavior space, allowing us to discover good joint behaviors in these high dimensional tightly-coupled scenarios. In contrast, we show that recent count-based exploration methods fail at scaling to these complex problems with high degrees of coupling.

II. Background and Related Work

The cooperative tightly coupled multi-robot search tasks can be represented as *Decentralized Partially-*

*This work was partially supported by the Air Force Office of Scientific Research grant No. FA9550-19-1-0195.

[1]Ayhan Alp Aydeniz and Kagan Tumer are with Collaborative Robotics and Intelligent Systems Institute, Oregon State University, Corvallis, Oregon, USA {aydeniza, kagan.tumer}@oregonstate.edu

[2]Enrico Marchesini is with Laboratory for Information & Decision Systems, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA e.marchesini@mit.edu

[3]Robert Loftin is with the Department of Computer Science, University of Sheffield, Sheffield, United Kingdom r.loftin@sheffield.ac.uk

Observable Markov Decision Processes (Dec-POMDPs) [17]. Formally, a Dec-POMDP is defined by a tuple $\langle N, \mathcal{S}, \mathcal{A}, P, \mathcal{O}, G, b_0, h, l, \pi \rangle$, where N is a finite set of agents, $\mathcal{S}$ is a finite set of states, $\mathcal{A} = \prod_{i \in N} \mathcal{A}_i$ is the set of *joint* actions and $\mathcal{O} = \prod_{i \in N} \mathcal{O}_i$ is the set of joint observations. To model the evolution of the system, $P(s^{t+1}|s^t, a^t)$ is the state transition probability function, while $O(o^t|s, a)$ is for observations. An episode ends after l steps and returns a global reward $\hat{G}$. In our setup, $\hat{G}$ is a sparse fitness and is available only at the end of l steps. Because the full state is not directly observed, agents only have access to individual observations o_i^t. Therefore, agent $i \in N$ keeps track of an action-observation history up to time t (denoted as $h_i^t = (o_i^0, a_i^0, \ldots, o_i^{t-1}, a_i^{t-1}, o_i^t)$), on which conditions its action selection process. As states, observations, and actions may all be stochastic, the fitness of a team is defined as an expectation over $\hat{G}$. The goal of a Dec-POMDP is to learn a joint policy $\pi = (\pi_1, \ldots, \pi_N)$ that maximizes this expectation.

A. Evolutionary Reinforcement Learning

Both RL and EAs have several benefits and drawbacks. RL agents employ gradient information to drive the learning process [18]; hence, they have a fast rate of convergence. However, gradient-based approaches are sensitive to hyper-parameters and suffer from high-variance and noisy estimates, which can lead to sub-optimal solutions [19], [20]. Conversely, EAs do have a slower rate of convergence, but their learning involves a population of solutions and a diversification process. Evolutionary Reinforcement Learning (ERL) [21], [22] was proposed to combine the benefits of both EAs and RL algorithms. Its EA enhances the diversity of the solutions generated during training and pushes the algorithms away from converging to sub-optimal solutions, while its RL component utilizes gradient-based algorithms to optimize specifically for the local objectives [23].

Despite the promising performance, ERL has received limited research attention in multiagent settings. As a leading solution among the few available approaches, Multiagent Evolutionary Reinforcement Learning (MERL) [11] considers individual agents that learn a local objective to accomplish the team-level coordination task. To leverage the benefits of ERL, MERL thus learns these local objectives using a RL algorithm, while leveraging an EA to learn team behaviors. However, as shown in previous works, MERL does not scale in the task complexity (*i.e.,* as multiagent tasks become more tightly-coupled), when agents have to learn using task-specific extrinsic objectives [24], [13]. Overall, it suffers from the issue of defined objectives [25], due to its EA being misdirected in the behavior space.

Specifically, in complex multiagent tasks, task-specific objectives lead the algorithm to converge to solutions that achieve well on the agent-level objectives but fail on the actual team objective. The work [13] proposes to use task-independent, intrinsic objective, but it does not scale to high dimensional domains. Though it achieves to avoid the mentioned misalignment of the objectives.

B. Exploration in Reinforcement Learning

A variety of solutions for enhancing exploration in RL settings have been investigated. For instance, some use prediction errors [26], [27], [28], [29], [30] and some employs novelty-oriented rewards [7], [31], [6]. In particular, the latter aims to enhance the diversity of the states encountered by "counting" states and rewarding agents based on the frequency of the visited states.

In discrete state spaces, these rewards are easy to formalize. For example, Novelty Seeking (NS) agents [13] have been employed in MERL, considering a simple discretization function that digitizes the values of the state vector to make them countable for continuous state spaces. However, in continuous settings, each state will likely be visited once (unless agents are allowed to not take any actions).[1] For high dimensional domains, the works [6], [7] employ density models to estimate a count, *pseudo-count*. However, when an agent exploits the novelty of a state, it receives a minimal reward for future visitations to that state, though it can be useful later in the environment. The paper [9] introduces a density function employing *k-nearest neighbor* search to measure the novelty of states and has shown remarkable results in single-agent settings, but the implementation requires a running average that can result in unstable reward values.

On the other hand, promoting state diversity as an entropy maximization problem has been recently investigated in RL. However, computing the entropy is intractable in high dimensional continuous domains, so different works propose to estimate such an entropy measure [32], [33], [34], [35], [10]. In more detail, [32], [33] maximize entropy based on the state density distribution, while [34], [35], [10] use a *k-nearest neighbor* estimate of entropy [14], [15]. Given the superior performance of the latter, in our work, we follow a similar approach to the works [35], [10].

C. Entropy Maximization in RL

Shannon [36] defined the entropy H of a probability density function (*pdf*) $f(x)$ over a random vector X as $H(X) = -\int f(x) \ln f(x)$.

Crucially, we can compute the entropy of any probability distribution. For example, in a RL context, the agent's probability distribution over the next possible actions at a state s has its policy entropy. Several Deep RL methods have employed this measure either in their objectives [37], [38], [39] or as a regularizer [40].

In contrast, we use entropy as a measure indicating the diversity of the states visited by an agent. Consider each state s_i as a random variable within the probability distribution over the state visitation history, h. We can thus assign probabilities based on the frequencies of each state in the history. All count-based reward functions can be then interpreted as maximizing this function, and the optimal distribution of states will be the maximum entropy distribution of states over

[1] Counting after quantization can be considered as binning the continuous states. However, in higher dimensional environments, this binning mechanism fails to maintain reliable visitation counts.

the history (*i.e.*, each state is visited only once in the history). However, as discussed above, relying on discretization or density models for estimating the state visitation can cause premature maximization of this function. In contrast, we will show the proposed *k-nearest neighbor* estimate allows a more accurate distribution of states.

III. Method

In contrast to the entropy maximization literature discussed in the previous section, the proposed SEM-MERL deals with the exploration-exploitation dilemma at two separate levels. Instead of trying to balance these components at both the agent and team-levels, we devote agent-level learning to learn fully exploratory policies, while exploiting the true team objective at the team-level. To this end, we utilize MERL as a learning framework where local policies are learned through RL algorithms and team policies are trained via an EA. Because MERL suffers from different convergence times of its learning modules, devoting RL agents to learn a fully exploratory behavior prevents RL agents from converging to sub-optimal solutions locally.

A. State Entropy Maximization

To compute an entropy, we do not have direct access to the *pdf* for a state distribution, but using the idea of estimating the entropy based on the sample spaces [15] has been useful in the RL settings [34], [35], [10]. So, we design our local reward based on the *k-nearest neighbor* estimation of the state entropy [14]. Say that $X_1, X_2, ..., X_n$ is a random sample from the distribution whose pdf is $f(x)$ having the entropy, $H(f)$.

The estimator of $H(f)$ can be represented as $\hat{H}(f) = -\frac{1}{n}\sum_{i=1}^{n}\log[f(\hat{X}_i)]$ where $\hat{f}$ is an estimator of the pdf $f(.)$.

k is an integer in the interval, $[1, n]$, and for $i = 1, \cdots, n$ and let $D_{i,k,n}$ be the Euclidean distance from X_i to its k^{th} neighbor, $||X_i - X_i^{k-nn}||_2$. According to Singh *et al.* [14], a good estimate of entropy, that can provide comparable estimates to the estimator proposed by Kozachenko and Leonenko [41], would be

$$\hat{H}_k^{(n)}(f) = \frac{1}{n}\log[\frac{n\pi^{p/2}D_{i,k,n}^p}{k\Gamma(p/2+1)}] + C_E \tag{1}$$

where C_E is the Euler constant: $C_E = -\int_0^\infty e^{-t}$, Γ is the gamma, p is the dimension of X, $\pi \sim 3.1415$. In essence, you can envision it as calculating the volumetric density of a sphere. Finally, we can use

$$\hat{H}_k^{(n)}(f) \propto \frac{1}{n}\sum_{i=1}^{n} logD_{i,k,n} \tag{2}$$

as an estimate of entropy derived by Beirlant *et al.* [15].

B. Reward Function Design

We provide our agents their local reward based on the function of Eq. 2 to enable them as local entropy maximizers.

In multiagent systems, search in a state space can be exhaustive and, to speed up exploration, we introduce salient events via our heuristic rewards, $V(h_i^t)$ [24], [13]. These heuristics are task-specific; however, in sparse reward domains where we can define salient events, they can be used for more efficient exploration [42]. These heuristics allows our agents to *highlight* the states that are worth visiting. Note that these heuristic rewards are not always available to the agents, they only appear, when agents experience salient events [13].

To compute the estimate of entropy as our reward function, we keep a dynamic history (*memory*) of observed states. After making an observation, we compute the Euclidean distance, D, of each observation in the history to the current observed state. Then, we sort distances of the observed states and derive our estimate of entropy over the *k-nearest neighbors* (where we use $k = 5$ for our experiments) as the reward by following the work [35]. k needs to be tuned according to the dimension of the state space, but, for our experiments, we keep it fixed to see how it behaves across varying sizes of state vectors. An alternative formulation can include a division of the distance, $D_{i,k,n}$, by the k, but our agents exhibit the best stability without this division.

So, the reward an agent receives at a state, s_i, becomes

$$r(s_i) = V(h_i^t)log(D_{i,k,n} + 1) \tag{3}$$

Algorithm 1: Computes a sequence of local entropy maximizing reward for agent i over a single episode.

```
Initialize state s^0, history h_i^0 = {o_i^0}.
for t ∈ [0, h − 1] do
    Retrieve action a_i^t, state s^{t+1} and observation o_i^{t+1}
    Compute Euclidean Distances, D, to o_i^{t+1} ∀o ∈ h_i^0
    Sort elements of D
    Choose the kth element
    reward ← V(h)log(D_{i,k,n} + 1) of Eq. 3
    h_i^{t+1} ← h_i^t ∪ {a_i^t, o_i^{t+1}}
    if V(h_i^{t+1}) > 1.0 then
        reward ← reward * V(h_i^{t+1})
    Yield reward
```

C. Multiagent Evolutionary Reinforcement Learning

MERL [11] is an ideal algorithm where agent-specific policies are trained on local objectives through gradient-based RL algorithms and team policies are trained based on a global objective via an EA. We define local objectives as fully exploratory objectives and EA learns exploitative policies. In this work, the local exploratory objective is the *k-nearest neighbor* estimate of state entropy (Section III-A).

In the MERL framework, its EA optimizes through a population of multi-head actors. Each head of a multi-head actor, representing an agent, is trained via its own replay buffer using an off-policy gradient-based RL algorithm, TD3 [43]. We employ a standard EA [44] as in the work [11].

Algorithm 2: Multiagent Evolutionary Reinforcement Learning (MERL) [11] with State Entropy Maximization

Initialize a population of M multi-head actor teams each having N agents, pop_π
Initialize a set of N replay buffers and N local TD3 agents
for $gen \in [1, \infty]$ **do**
  foreach *team* $\pi \in pop_\pi$ **do**
    $Fitness = 0$
    for $t \in [0, Timesteps]$ **do**
      foreach $agent\ A \in A_1, \ldots, A_N$ **do**
        Compute local reward, r^t, (via Algorithm 1)
        foreach *ReplayBuffer* $R \in R_1, \ldots, R_N$ **do**
          append (o^t, o^{t+1}, a^t, r^t) to R
    Compute team reward G
    Assign G as $Reward$ of $team\ \pi$
  // Evolve pop_π
  Return the $Champion$
  Send the policy gradient team to pop_π replace with the team achieving the lowest reward

IV. Experiments

Our experiments aim to answer the following questions:

- How well do our agents perform in high dimensional state spaces?
- Are our agents able to discover good behaviors in tightly coupled tasks?
- How do our agents perform tasks complicated by both tight-coupling and high dimensional state spaces?

We test our agents in a well-established multi-robot coordination and navigation domain called *multi-rover exploration* [16], since navigation is a well-established domain also in many single-agent RL fields (*e.g.*, safety [45], [46]). This domain requires a team of agents to solve a tightly coupled multi-robot problem where multiple agents need to take correct closely coordination joint actions. As this number of agents increases the task complexity increases. Therefore, it is difficult to unearth these joint actions. When the dimension of their state space increases, it gets much more challenging to discover good team behaviors. Thus, we test our state entropy maximizing agents in scenarios where we increase the state-vector size and the task complexity.

A. Multi-Rover Exploration Domain

Multi-rover exploration domain [16], [11] (Fig. 2) is a tightly coupled multiagent domain where there are multiple rovers that need to observe multiple points of interest (POIs). Each of these POIs needs to be observed simultaneously by a number of rovers which is determined according to the coupling factor in the environment. However, robots first need to discover these POIs by navigating in the environment. The coupling factor is not known to the agents and agents can only infer about it when they receive their rewards, but these rewards are not provided to them until the end of an episode. And, they can only achieve an increase in the reward, when they successfully observe a POI with their teammates. These rovers are equipped with two types of sensors, POI and rover sensors. In an environmental configuration, there exists a parameter setting the resolution of how agents perceive their environment. This resolution determines the number of sensors that an agent has. For example, 90° of resolution denotes the case where an agent has 4 pairs of sensors (4 90° channels in a 360° of coverage). From each channel, agents can detect both their teammates and POIs.

POI and rover sensors' values for an agent, A_i, are computed as:

$$s_{POI} = max(\frac{W_{POIj}}{\delta(A_i, POIj)}), s_{rover} = max(\frac{1}{\delta(A_i, A_j)}) \tag{4}$$

where W_{POIj} is the worth of a POI_j detected by the sensors of an agent A_i, and $\delta(.,.)$ denotes the distance between two objects. Agents take actions defined by two continuous values, speed and direction.

To observe a POI, agents need to be present within POIs' activation radius. We introduce the heuristic reward, $V(h_i^t)$, of Eq. 3, when agents navigate into the activation radius of a POI. The value of saliency is equal to the value of that POI. Agents are not provided this radius, but when they navigate to a POI and simultaneously observe that POI with a number of their teammates, they achieve a global team reward defined by the global reward function, G:

$$G = \frac{\sum_{k=1}^{L} W_{POI_k} I(POI_k)}{\sum_{i=1}^{L} W_{POI_j}} \tag{5}$$

where L is the number of POIs, W_{POI_k} is the value of the POI_k, and $I(.)$ is the Boolean function returning 1, if POI_k is visited by a sufficient number of agents, 0, otherwise.

B. Experimental Parameters

We compare our proposed method with novelty seeking (NS) agents [24], [13] that employ quantization to make continuous states countable. It discretizes the values of the local observations of agents. As the work [13] suggests, we use binary quantization of the values, and allows us to construct the reward function, $\frac{1}{count(o)}$, for an observation o. The drawback of this function is that it does not allow us to decrease the *quantization level*, because the lowest possible resolution for an element is 1-bit. Our proposed method utilizes the estimate of state entropy, whereas NS agents derive a count based reward. NS agents were proposed to balance exploration and exploitation at the hierarchical levels as well; however, its main module that derives its count-based reward is not scalable to high dimensional state spaces. Because quantization is applied only at value-level, NS-MERL does not apply any modifications to the actual dimension of agents' state-vectors.

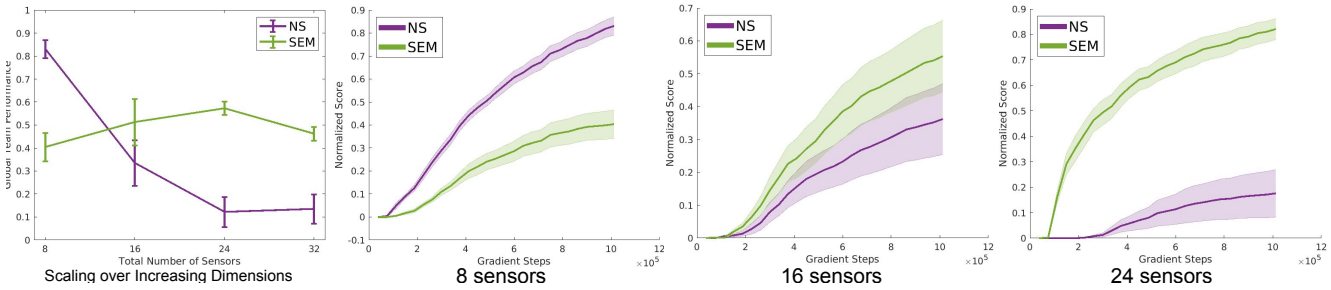


Fig. 1: Performances of Novelty Seeking (NS) and State Entropy Maximizing (SEM) agents in the experimental setup where we vary the dimension of state space - 8 agents with a constant coupling of 5, Densely Distributed 9 POIs on three layers and the episode length is 50. Each sensor configuration represents a total number of POI and rover sensors. The inner-most layer has the POIs with the lowest values, the outer-most layer has the highest values on a 20x20 map.

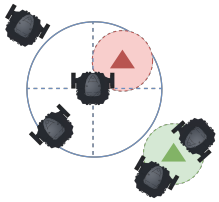
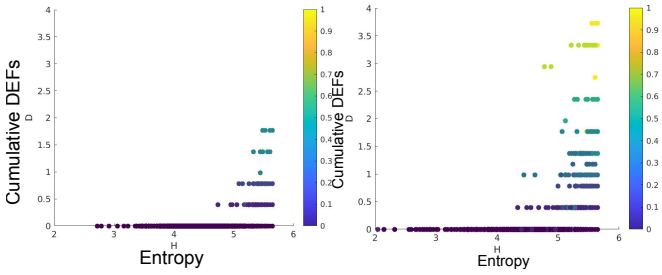


Fig. 2: Multi-Rover environment with a coupling of 2. The blue lines represent the field view and observations of a rover. The red triangle is an unobserved POI, as only one rover is inside its activation area. In contrast, the green triangle is successfully observed by two rovers simultaneously.



(a) Novelty Seeking (b) State Entropy Maximizing

Fig. 3: Analysis of the behaviors learned in Fig. 4 when coupling is 7. Each behavior is defined via their contribution to the team that they are in using Difference Evaluation Functions (DEFs) [16], D, and their state entropy, H. The color-gradient denotes their team's total reward.

In our multi-robot coordination domain, we test our method against NS agents first in a configuration where we increase the dimension of the state space by increasing the number of sensors of agents. Secondly, we test our agents to answer how they perform in tightly coupled tasks. Because NS agents were presented as a method that can explore efficiently under tight-coupling, we test our agents also under increasing coupling factor. Our hypothesis is that, as the number of dimensions of a multiagent state space increases, NS agents fail or poorly perform in tightly coupled tasks, but, in contrast, our proposed method addresses this problem.

Note that we do not only increase the state space to compute our reward, but the states given to the neural networks also grow at the same rate; therefore, as we add additional sensors, increase the dimension of the problem.

C. Results

The performances on plots are the results of 135 evolutionary generations and the RL steps (gradient updates) are provided on the x-axes. Each plot denotes the average of 10 statistical runs. We employ the same hyper-parameters provided in the works [13], [11], [10].

SEM-MERL aims to enable agents to learn exploratory behaviors in high dimensional tasks. In this section, we test our agents mainly in two experimental configurations and, then analyze the behaviors learned by our agents.

In all experimental configurations, we deploy 8 rovers in an environment where there are 9 POIs. These POIs are distributed according to their values on three layers and these layers are parallel to each other where each represents 60° chord of a circle. Each layer has 3 POIs. The POIs that are worth the least are on the closest layers and, as agents move away from their starting point, they discover POIs with higher values. The most inner layer has the POIs worth 2, the middle one has the POIs valued 5, and the outer one has the POIs with value 10. We expect agents to learn behaviors that discover POIs with higher values. However, it is impossible for agents to observe all POIs (due to limited episode length); hence, we normalize the performances on each plot. The leftmost plot, in Fig. 1, is normalized according to the maximum performance of all runs of its experiments.

1) Growing State Space: In our first set of experiments, we set a constant coupling factor, 5, and varied the dimensionality of state spaces. In Fig. 1, when agents have 8 total sensors (4 POI and 4 rover sensors), agents using NS agents outperform our agents (under the constant k of 5). However, as soon as we grow the state space, our agents outperform NS agents. In these plots, as we increase the number of dimensions of agents' state vectors, we see that our SEM agents scale much better than NS agents.

2) Increasing Task Complexity: Increasing the coupling factor, requiring more agents to successfully observe a POI,

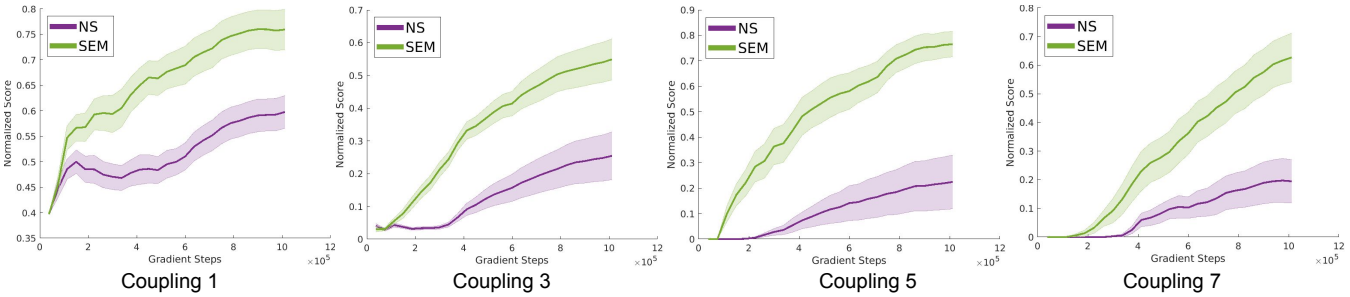


Fig. 4: Performances of Novelty Seeking (NS) and State Entropy Maximizing (SEM) agents in the experimental setup where we vary the coupling factor - 8 agents with a constant dimension of the state space with 32 sensors (16 POI, 16 rover sensors), Densely Distributed 9 POIs on three layers and the episode length is 50. Each sensor configuration represents a total number of POI and rover sensors. The inner-most layer has the POIs with the lowest values, the outer-most layer has the highest values on a 20x20 map.

makes finding of good team behaviors challenging. It basically squeezes good team policies down to narrow areas in the policy-space. NS agents were proposed to show that novelty-seeking in multiagent systems helps agents find those policies even under high coupling factors. However, growing the state space of agents moves the problem to a different level and makes the discovery of the team policies where agents take highly coordinated actions much more difficult.

When agents are equipped with 32 sensors, SEM-MERL agents are able to discover good joint actions and they outperform NS agents across all coupling factors (Fig. 4).

3) Behavioral Diversity: We expect our agents to perform in both tightly coupled and high dimensional tasks through their ability to learn a large range of behavioral diversity. The paper [13] has already shown that employing these exploration-driven rewards enriches the behavioral diversity, so that agents are able to perform even under highly tightly coupled tasks. In this paper, we multiagent-specific diversity plot, proposed in the work [13], representing each behavior on three main axes. Fig. 3 has two plots whose x-axes represent a behavior's state entropy, H, (computed over discretized versions of visited states) y-axes denote a behavior's contribution, D, to its team's global reward [16] and the third axes use the color gradient where warmer colors represent a behavior's team's reward, G.

V. Discussion

Our method is broadly applicable to hardware because it does not depend on any specific type of input, but rather focuses on measuring distances of state observations (*e.g.* LIDAR data, images) which can be adopted by any multi-robot system. After generating trajectories, the robots can query alternate paths in physical robots or on any ROS-based simulator employing Gazebo. We will consider the porting on real hardware as future work.

This paper addresses the problem of poor exploration of multiagent teams operating in high dimensional state spaces. Instead of the conventional way of addressing exploration-exploitation dilemma, we divide the main problem, exploring at the agent-level and exploiting at the global team level. We use *k-nearest neighbor* estimate of state entropy and maximize this estimate locally by providing it to the agents as rewards. Because we compute the reward function as *k-nearest neighbor* estimate of entropy, we do not rely on density models or discretization of state vectors, and our method is able to promote maximum entropy state distribution while avoiding the convergence of the novelty of states to zero.

We test our SEM agents against Novelty Seeking (NS) agents [24], [13] in a cooperative multi-robot coordination domain where agents are dependent on each other's actions. As this degree of dependency (coupling) increases, the difficulty of tasks increases and finding good team behaviors becomes difficult. Even in the state spaces with increased dimensionality, our agents are able to learn good team behaviors (Fig. 4) under high coupling. From our behavior analysis (Fig. 3), agents are able to do this by learning behaviors that do not only have higher state entropy, but also higher contribution to their team's reward.

VI. Conclusion

Overall, we apply the strategy of *explore locally and exploit globally* to high dimensional multiagent state spaces. To our knowledge, this paper is making the contribution of proposal of a framework that significantly improves the performance and enables the exploration of closely coordinated team policies of tightly coupled multiagent systems operating in high dimensional state spaces. Our work also enables agents to compute a reward function in when the state vectors have elements varying differently (*e.g.* velocity, distances, positions). When the computation is based on discretization of states like NS rewards, discretization should be carefully hand-tuned and this makes their scalability to different environments more difficult.

The current work is leading the way toward an efficient exploration framework in high dimensional state spaces. As a future work, we suggest that exploration via joint states of agents can improve the long-term learning. However, the computation of Euclidean distances can be uninformative after a certain dimension of vectors. We recommend using an embedding network [9], or an encoder [10] to reduce the dimension of state spaces that are larger than 32 dimensions down to 24, or 32 dimensions, as our experiments suggest.

REFERENCES

[1] L. Yliniemi, A. K. Agogino, and K. Tumer, "Multirobot coordination for space exploration," *AI Magazine*, vol. 35, no. 4, pp. 61–74, 2014.

[2] Z. Zhou, J. Liu, and J. Yu, "A survey of underwater multi-robot systems," *IEEE/CAA Journal of Automatica Sinica*, vol. 9, no. 1, pp. 1–18, 2021.

[3] E. Marchesini and A. Farinelli, "Centralizing state-values in dueling networks for multi-robot reinforcement learning mapless navigation," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 4583–4588.

[4] ——, "Enhancing deep reinforcement learning approaches for multi-robot navigation via single-robot evolutionary policy search," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 5525–5531.

[5] Ö. Şimşek and A. G. Barto, "An intrinsic reward mechanism for efficient exploration," in *Proceedings of the 23rd international conference on Machine learning*, 2006, pp. 833–840.

[6] M. Bellemare, S. Srinivasan, G. Ostrovski, T. Schaul, D. Saxton, and R. Munos, "Unifying count-based exploration and intrinsic motivation," *Advances in neural information processing systems*, vol. 29, 2016.

[7] H. Tang, R. Houthooft, D. Foote, A. Stooke, O. Xi Chen, Y. Duan, J. Schulman, F. DeTurck, and P. Abbeel, "# exploration: A study of count-based exploration for deep reinforcement learning," *Advances in neural information processing systems*, vol. 30, 2017.

[8] J. Fu, J. Co-Reyes, and S. Levine, "Ex2: Exploration with exemplar models for deep reinforcement learning," *Advances in neural information processing systems*, vol. 30, 2017.

[9] A. P. Badia, P. Sprechmann, A. Vitvitskyi, D. Guo, B. Piot, S. Kapturowski, O. Tieleman, M. Arjovsky, A. Pritzel, A. Bolt *et al.*, "Never give up: Learning directed exploration strategies," *arXiv preprint arXiv:2002.06038*, 2020.

[10] Y. Seo, L. Chen, J. Shin, H. Lee, P. Abbeel, and K. Lee, "State entropy maximization with random encoders for efficient exploration," in *International Conference on Machine Learning*. PMLR, 2021, pp. 9443–9454.

[11] S. Majumdar, S. Khadka, S. Miret, S. McAleer, and K. Tumer, "Evolutionary reinforcement learning for sample-efficient multiagent coordination," in *International Conference on Machine Learning*. PMLR, 2020, pp. 6651–6660.

[12] D. H. Wolpert and K. Tumer, "Optimal payoff functions for members of collectives," *Advances in Complex Systems*, vol. 4, no. 02n03, pp. 265–279, 2001.

[13] A. A. Aydeniz, R. Loftin, and K. Tumer, "Novelty seeking multiagent evolutionary reinforcement learning," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2023, pp. 402–410.

[14] H. Singh, N. Misra, V. Hnizdo, A. Fedorowicz, and E. Demchuk, "Nearest neighbor estimates of entropy," *American journal of mathematical and management sciences*, vol. 23, no. 3-4, pp. 301–321, 2003.

[15] J. Beirlant, E. J. Dudewicz, L. Györfi, E. C. Van der Meulen *et al.*, "Nonparametric entropy estimation: An overview," *International Journal of Mathematical and Statistical Sciences*, pp. 17–39, 1997.

[16] A. K. Agogino and K. Tumer, "Unifying temporal and structural credit assignment problems," in *Autonomous Agents and Multi-Agent Systems Conference*, 2004.

[17] F. A. Oliehoek and C. Amato, *A concise introduction to decentralized POMDPs*. Springer, 2016.

[18] E. Marchesini and C. Amato, "Improving deep policy gradients with value function search," in *The Eleventh International Conference on Learning Representations*, 2023. [Online]. Available: https://openreview.net/forum?id=6qZC7pfenQm

[19] E. Marchesini, D. Corsi, and A. Farinelli, "Genetic soft updates for policy evolution in deep reinforcement learning," in *International Conference on Learning Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=TGFO0DbD_pk

[20] L. Marzari, D. Corsi, E. Marchesini, and A. Farinelli, "Curriculum learning for safe mapless navigation," in *Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing*, 2022.

[21] S. Khadka and K. Tumer, "Evolution-guided policy gradient in reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[22] E. Marchesini, D. Corsi, and A. Farinelli, "Exploring safer behaviors for deep reinforcement learning," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 7, pp. 7701–7709, Jun. 2022.

[23] E. Marchesini and A. Farinelli, "Genetic deep reinforcement learning for mapless navigation," in *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, 2020, p. 1919–1921.

[24] A. A. Aydeniz, A. Nickelson, and K. Tumer, "Entropy-based local fitnesses for evolutionary multiagent systems," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, 2022, pp. 212–215.

[25] J. Lehman and K. O. Stanley, "Novelty search and the problem with objectives," *Genetic programming theory and practice IX*, pp. 37–56, 2011.

[26] R. Houthooft, X. Chen, Y. Duan, J. Schulman, F. De Turck, and P. Abbeel, "Vime: Variational information maximizing exploration," *Advances in neural information processing systems*, vol. 29, 2016.

[27] J. Oh, X. Guo, H. Lee, R. L. Lewis, and S. Singh, "Action-conditional video prediction using deep networks in atari games," *Advances in neural information processing systems*, vol. 28, 2015.

[28] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell, "Curiosity-driven exploration by self-supervised prediction," in *International conference on machine learning*. PMLR, 2017, pp. 2778–2787.

[29] B. C. Stadie, S. Levine, and P. Abbeel, "Incentivizing exploration in reinforcement learning with deep predictive models," *arXiv preprint arXiv:1507.00814*, 2015.

[30] A. Ecoffet, J. Huizinga, J. Lehman, K. O. Stanley, and J. Clune, "Go-explore: a new approach for hard-exploration problems," *arXiv preprint arXiv:1901.10995*, 2019.

[31] N. Haber, D. Mrowca, S. Wang, L. F. Fei-Fei, and D. L. Yamins, "Learning to play with intrinsically-motivated, self-aware agents," *Advances in neural information processing systems*, vol. 31, 2018.

[32] L. Lee, B. Eysenbach, E. Parisotto, E. Xing, S. Levine, and R. Salakhutdinov, "Efficient exploration via state marginal matching," *arXiv preprint arXiv:1906.05274*, 2019.

[33] E. Hazan, S. Kakade, K. Singh, and A. Van Soest, "Provably efficient maximum entropy exploration," in *International Conference on Machine Learning*. PMLR, 2019, pp. 2681–2691.

[34] M. Mutti, L. Pratissoli, and M. Restelli, "Task-agnostic exploration via policy gradient of a non-parametric state entropy estimate," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 10, 2021, pp. 9028–9036.

[35] H. Liu and P. Abbeel, "Behavior from the void: Unsupervised active pre-training," *Advances in Neural Information Processing Systems*, vol. 34, pp. 18 459–18 473, 2021.

[36] C. E. Shannon, "A mathematical theory of communication," *The Bell system technical journal*, vol. 27, no. 3, pp. 379–423, 1948.

[37] C. F. Lee and D. H. Wolpert, "Product distribution theory for control of multi-agent systems," in *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems, 2004. AAMAS 2004.*, 2004.

[38] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International conference on machine learning*. PMLR, 2018, pp. 1861–1870.

[39] B. Eysenbach, A. Gupta, J. Ibarz, and S. Levine, "Diversity is all you need: Learning skills without a reward function," *arXiv preprint arXiv:1802.06070*, 2018.

[40] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," in *arXiv*, 2017.

[41] L. F. Kozachenko and N. N. Leonenko, "Sample estimate of the entropy of a random vector," *Problemy Peredachi Informatsii*, vol. 23, no. 2, pp. 9–16, 1987.

[42] C.-A. Cheng, A. Kolobov, and A. Swaminathan, "Heuristic-guided reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 13 550–13 563, 2021.

[43] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *International conference on machine learning*. PMLR, 2018, pp. 1587–1596.

[44] D. B. Fogel, *Evolutionary computation: toward a new philosophy of machine intelligence*. John Wiley & Sons, 2006, vol. 1.

[45] E. Marchesini, L. Marzari, A. Farinelli, and C. Amato, "Safe deep reinforcement learning by verifying task-level properties," in *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*, 2023, p. 1466–1475.

[46] L. Marzari, E. Marchesini, and A. Farinelli, "Online safety property collection and refinement for safe deep reinforcement learning in map-

less navigation," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 7133–7139.