Balancing Teams with Quality-Diversity for Heterogeneous Multiagent Coordination

Gaurav Dixit
Oregon State University
Corvallis, USA
dixitg@oregonstate.edu

Kagan Tumer
Oregon State University
Corvallis, USA
kagantumer@oregonstate.edu

ABSTRACT

Many real world multiagent applications such as search and rescue or traffic management require coordinating teams of heterogeneous agents. Unfortunately, learning is difficult in such domains as agents often converge to a limited set of "acceptable" behaviors that may be suboptimal. Quality-Diversity methods offer to alleviate this problem by shifting the focus from optimizing behaviors to finding a diverse repertoire of behaviors. However, in multiagent environments with diverse and tightly-coupled tasks, exploring the entire space of behaviors is often intractable. Agents must focus on only finding useful behaviors that are conducive to good team performance. We introduce Behavior Exploration for Heterogeneous Teams (BEHT), a multi-level training framework that allows systematic exploration of the agents' behavior space required to complete diverse tasks as a coordinated team. Via a combination of diversity search using dense agent-specific rewards and team-objective maximization via an evolutionary method, agents' behavior space is relearned iteratively to find diverse cooperative behaviors. In multiagent environments which call for diverse coordinated team behaviors, we show that BEHT allows agents to learn diverse synergies that are demonstrated by the diversity of acquired agent behavior in response to the environment and other heterogeneous agents.

KEYWORDS

Adaptive Team Balancing, Quality Diversity, Multiagent learning, Evolution

ACM Reference Format:
Gaurav Dixit and Kagan Tumer. 2022. Balancing Teams with Quality-Diversity for Heterogeneous Multiagent Coordination. In *Proc. of the 21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2022), Auckland, New Zealand, May 9–13, 2022*, IFAAMAS, 9 pages.

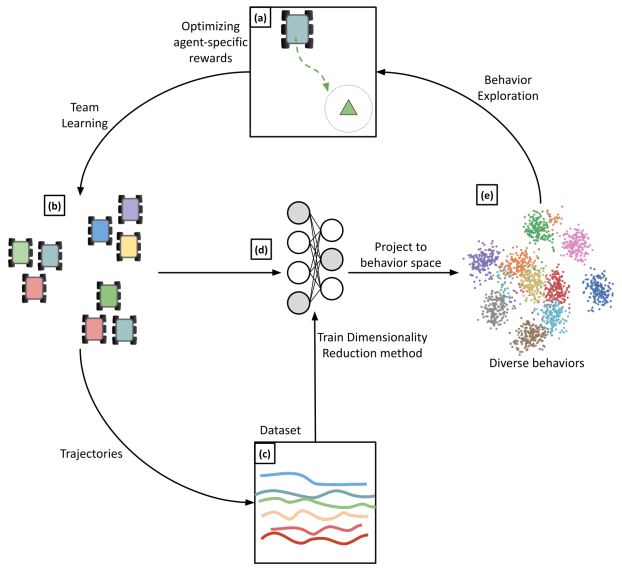
(a)
Optimizing agent-specific rewards
Behavior Exploration
Team Learning
(b)
(d)
Project to behavior space
(e)
Diverse behaviors
Train Dimensionality Reduction method
Dataset
Trajectories
(c)

Figure 1: BEHT: A multi-level training method to explore diverse agent behaviors for heterogeneous team coordination. While agents explore their behavior space using dense agent-specific rewards (sub-figure (a)), an evolutionary method retains behaviors that optimize the team-based objective (sub-figures (b), (c)). This biased selection allows iterative transformation of the agent behavior space by applying a dimensionality reduction algorithm to the agent trajectories (sub-figure (d)). The resulting method allows heterogeneous agents to progressively explore regions of the behavior space that promote team coordination on diverse goals.

1 INTRODUCTION

Multiagent systems are indispensable in complex real-world tasks that require distributed control and robustness, and have been applied to a wide variety of tasks such as search and rescue [31], air traffic control [13, 28], bandwidth management [2, 10] and satellite configuration [8]. Interestingly, many such real-world applications require agents to work together as a team.

In spite of the successful applications, learning in heterogeneous multiagent systems remains difficult, which limits their wide-scale adoption [29]. An important aspect of agents in the heterogeneous setting is the ability to reason about the behavior of other agents and adapt [1]. Specialization of behavior, often complementary, is necessary to allow working on diverse tasks and satisfy hardware and morphology restrictions. This presents a need to systematically find useful specializations, execute diverse behaviors and assume new roles in response to the capabilities of other agents to successfully solve the task as a team. In environments with diverse tasks and heterogeneous agents, either the types of agents and the required capabilities must be pre-designed by an expert or they must be explored whilst learning.

Quality Diversity (QD) methods are a new family of evolutionary methods that favour functional diversity of behaviors over an

Proc. of the 21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2022), P. Faliszewski, V. Mascardi, C. Pelachaud, M.E. Taylor (eds.), May 9–13, 2022, Auckland, New Zealand. © 2022 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). All rights reserved.

objective measure of performance [3, 17, 23]. By evolving a repertoire of behaviors, QD methods enable an agent to operate with several high performing and diverse behaviors [21]. In the context of heterogeneous multiagent teams, QD methods are appealing due to their emphasis on diversity that can aid in agent specialization and ad-hoc cooperation.

In multiagent settings however, exploring the space of all possible behaviors is often intractable due to the potentially large behavior space imposed by multiagent interaction. Furthermore, in problems involving diverse tasks, agents cannot be expected to be able to perform the entire spectrum of behaviors required to satisfy the tasks and must specialize. If useful behaviors are discovered by different agents to specialize, the number of agents with the different behaviors must also be balanced so as to maximize the team objective.

In this work, we introduce Behavior Exploration for Heterogeneous Teams (BEHT), a multi-level training framework that allows systematic exploration of the agent capabilities required to complete diverse tasks as a coordinated team. The two primary goals, behavioral diversity search and optimization of team objectives, are decoupled into two optimization processes. A gradient-based optimizer trains a population of policies to optimize agent-specific rewards for useful primitive behaviors. Policies are projected in a behavior space that is inferred by applying a dimensionality reduction method to the policy trajectories. By limiting the number of policies in a given radius of the behavior space, diversity in policies in encouraged.

The behavioral diversity search is followed by an evolutionary algorithm (EA) that maximizes the sparse team-based objective. The selection mechanism of the EA acts as a filter over the regions of the behavior space that contribute to the team objective. The trajectories of the policies from the filtered behavior space are then used to relearn a new behavior space that captures the variance of the current agent behaviors and allows moving in the direction of diversity that is supported by the team objective. *The key insight of this work is: the decoupling of diversity search and optimization, with iterative refinement of the behavior space, allows the transformation of fitness on the team objective to the direction of functional diversity.*

The primary contribution of this work is to introduce BEHT, a multi-level quality diversity and optimization method for robust adaptation and coordination in heterogeneous agent teams. Figure 1 outlines BEHT's iterative learning approach. We demonstrate the strength of BEHT on a multiagent rover exploration problem with sparse feedback and a diverse set of goals that requires tight agent coupling between heterogeneous agents. Our approach shows significant performance improvement and adaptibility in multiagent settings over Malthusian Reinforcement Learning [18] and the Intrinsic Curiosity module [22], current state-of-the-art diversity search methods.

2 BACKGROUND

This section provides an overview of recent work in multiagent learning, Quality Diversity methods and ad-hoc teaming.

2.1 Multiagent Learning

Learning in a multiagent setting is inherently difficult compared to single agent learning because the learning agents modify the "environment" in which all other agents learn and operate [4]. Single agent learning methods are not readily applicable to multiagent scenarios since the underlying Markov assumptions of these methods are violated [16, 29]. This is a result of the environment being non-stationary as each agent learns independently and considers the other learners as part of the environment. Independent learners (e.g., Independent Q-Learning) ignore the multiagent nature of the problem entirely and fail when other agents in the system (especially opponents) adapt their policy. Despite the brittleness, they have been used in practice when scalability and robustness are the primary goals [19, 27].

Majority of the most effective multiagent learning techniques typically rely on communication, agent modeling or shared fitness for learning simultaneously with other agents [11]. Reward or fitness shaping has also been effective in enabling agents to learn locally, while ensuring they still optimise the system level objectives [24]

In tightly coupled multiagent learning, training based solely on the team rewards is often difficult as they require agents to take specific joint actions, and are hence inherently sparse. Moreover, relying on agent-specific rewards that incentivize learning primitive behaviors is sub-optimal since they can either be misaligned with or fail to capture the team objective. Multiagent Evolutionary Reinforcement Learning (MERL) [14], a cooperative multiagent reinforcement learning approach utilizes a bi-level optimization of sparse team rewards and dense agent specific rewards to ensure alignment and faster convergence. However, extending MERL to heterogeneous agent teams is non-trivial since MERL does not leave room for explicit diversity search.

2.2 Quality Diversity

Quality Diversity (QD) methods have been widely applied in robotics and multiobjective optimization problems for learning a repertoire of diverse behaviors [3, 5, 21]. While traditional learning is geared toward optimization of controls, QD shifts the focus to learning a wide range of novel behaviors that can solve the task. Controllers are typically defined by parametric functions that can be optimized via evolutionary or machine learning approaches. Novelty is measured by condensing the learned behavior to a compact representation called a Behavior Characterization (BC). The BC can either be hand-designed or learned using dimensionality reduction methods like deep auto-encoders [12]. Methods like the intrinsic curiosity module are also suitable for diversity search [22]. They differ from QD methods in that their primary goal is exploration for maximization of a reward, whereas QD methods primarily disregard reward maximization to explicitly favor diversity maximization.

In practice, most variants of the QD algorithm such as Novelty Search and MAP-Elites [17, 20] can be described as iterative processes with two primary operations: 1) Organize a collection of behaviors along their BC; and 2) Within a population of behaviors, select the most novel behaviors that have the highest fitness within their niche. The organization of behaviors can either be a structured grid or an unstructured archive.

2.3 Multiagent Ad-Hoc Teaming

The goal of ad-hoc teaming is to create robust agents that can perform successfully in teams that they have not participated in a priori [26]. The ability to adapt to the behavior of team members is particularly crucial for real world applications, since they often involve open multiagent settings where agents are free to enter and exit the system. When agents have unique capabilities, their ability to become part of ad-hoc teams can fundamentally change the outcome in cooperative tasks. This would entail that the agents can cooperate toward common goals without having learned to do so during training. To cooperate, agents must not only have aligned goals, but the ability to predict and model others' reactions to their behavior. Learning transferable skills that can be adapted, reused and aligned with unseen team members has shown success [7].

While the behavior can be modeled when agents are similar, diverse behaviors and specialization can complicate this significantly, especially if it is unclear to the agent what others' behavior sets are. Learning diverse behaviors and adapting to changing teams is a cornerstone of successful team performance. Recently, Malthusian Reinforcement Learning has been successfully shown to create synergistic teams in which agents specialize and operate with other agents in different environments [18]. By operating simultaneously in different teams on different environments, agents learn to discover unique strategies to outcompete other agents on a shared resource. However, in absence of adversarial pressure created by shared resources, agents have no incentive to discover diverse behaviors.

3 METHOD: BEHAVIOR EXPLORATION FOR HETEROGENEOUS TEAMS (BEHT)

BEHT starts with a population of randomly initialized policies. The policies are trained using Proximal Policy Optimization (PPO), a gradient based optimization method, on dense agent-specific rewards [25]. The trajectories of the trained policies (any data related to the policies that captures their behavior can be used) are collected as the initial dataset. The dimensionality reduction algorithm is trained on this dataset to learn a latent representation of the trajectories. This low dimentional representation defines the behavior space of the policies. Each policy from the population is then projected in the behavior space. This completes the initialization of BEHT.

Diversity search: With the agent policies projected in the behavior space, a QD like iteration is performed:

(1) A policy is randomly selected from the population;
(2) The policy is mutated by perturbing weights of the policy network probabilistically;
(3) The mutated policy is trained using PPO to maximize a dense agent-specific reward for a fixed number of episodes and evaluated;
(4) The trained policy is finally projected in the behavior space.

Algorithm 1 provides pseudo-code for these steps. There are three key differences that set this apart from a standard QD iteration [5]. Firstly, instead of mutating a trajectory, a policy is directly mutated. Secondly, every mutation is followed by training on a dense agent-specific reward. Finally, unlike QD, the behavior descriptor used to project the policies in the behavior space is automatically determined by feeding the trajectory of the policies to a dimensionality reduction method. Over the course of several iterations of this process, the mutations allow to progressively fill the behavior space.

Algorithm 1: Diversity Search

```
1  Function policy_qd($pop_\pi$):
       Input: population of policies, $pop_\pi$
       Result: repertoire of diverse policies
2      for $n \leftarrow 0$ to $I$ do
3          $\pi$ = selection($pop_\pi$)  // random policy from population
4          $\pi\prime$ = mutate($\pi$)
5          $\pi\prime \leftarrow$ maximize($reward_{agent}$)  // train with gradient
            method on agent-specific reward
6          $bc\prime$ = evaluate_($\pi\prime$)
7          $bc \leftarrow$ closest_policy_in_archive()
8          if distance ($bc\prime$, $bc$) > $\lambda$ then
               /* new policy occupies unoccupied region in
                  behavior space */
9              add_policy ($\pi\prime$, $bc\prime$)
10         else
               /* new policy competes locally with the closest
                  policy in the behavior space */
11             $\pi_c$ = policy($bc\prime$)
12             if reward($\pi_c$) < reward($\pi\prime$) then
13                 add_policy ($\pi\prime$, $bc\prime$)
14                 remove_policy ($\pi_c$)
```

Team-objective optimization: The next phase of BEHT uses an evolutionary algorithm (EA) to train the policies on the sparse team-based objective. A population of teams, each with equal number of random policies from the policy population, is initialized. A policy that is part of multiple teams uses the name network and replay buffer across teams. The population of teams is then evaluated on the team task and a team reward for each team is distributed at the end of the episode as its fitness. A selection operator selects a portion of the population for survival with probability proportional to their fitness. The weights of the teams are probabilistically perturbed through mutation and crossover operators to create the next generation of teams. A portion of the teams with the highest fitness is preserved at each iteration.

After several evolutionary updates, only a portion of the policy population with the highest fitness (the summed fitness of the teams in which the policies participated) will be retained. The EA thus essentially filters out regions of the behavior space that have the highest fitness on the team objective. By means of adjusting the population dynamics of teams and policies within the teams via fitness, the EA also balances the number and quantity of different behaviors needed for optimal team performance.

Behavior space refinement: Finally, the latent representation of the policies, the behavior space, is updated to take into account the new behaviors discovered from mutation of policies on agent-specific rewards and the updated distribution of those policies in the behavior space. By applying the dimensionality reduction method to the updated policies, the behavior space is relearned to align with the maximum variance (diversity) of the highest performing policies. The policies are then projected back in the updated behavior space. Often, this update will project originally distant policies closer together, in which case, the policy with the highest agent-specific reward will be retained. This is a safe replacement since closeness in the behavior space implies that the policies in consideration behaved similarly in teams. The size of the policy population is thus often reduced after a dimensionality reduction update, but will expand again in the next QD-like iteration. BEHT then goes back to the QD-like phase of exploration and agent-specific reward maximization while exploiting the experiences collected via the evolutionary phase. The QD-like phase can now continue to explore behavioral diversity in the new behavior space which likely offers regions that were not already fully covered in the previous iteration.

BEHT iteratively performs the following three phases:

(1) **Diversity search:** Filling the behavior space by maximizing agent-specific reward;
(2) **Team-objective optimization:** Evolution of teams of policies to maximize the team objective and balance proportions of behaviors;
(3) **Behavior space refinement:** Learning a new behavior space by feeding updated policy trajectories to the dimensionality reduction method.

Algorithm 2 provides a sketch of this iterative process.

Algorithm 2: Multiagent Coordination

```
Function train_teams(K:Integer, N:Integer):
    Initialize a population $pop_\pi$ of N policy networks $\pi$
      with weights $\theta$
    Initialize N empty replay buffers, one for each policy
      network
    for $iteration \leftarrow 0$ to $\infty$ do
        /* Fill behavior space using QD */
        $pop_\pi$, data $\leftarrow$ policy_qd ($pop_\pi$)
        $train\_dr$ (data)
        $project\_policies()$
        Initialize a population $pop_K$ of K teams
        for $generation \leftarrow 0$ to $G$ do
            foreach $team\ p \in pop_K$ do
                Assign M random policies $\pi$ from $pop_\pi$
                $fitness_k$ = evaluate ($p$)
            Rank population $pop_K$ based on fitness
            Select the first $e$ teams $\in pop_K$ as elites
            Select the remaining $(M - e)$ teams from $pop_K$,
              to form set $S$ using tournament selection
            while $|S| < (M - e)$ do
                crossover between randomly sampled policy
                  $\pi \in e$ and $\pi \in S$ and append to S
        $pop_\pi \leftarrow S \cup e$
        $project\_policies()$
```

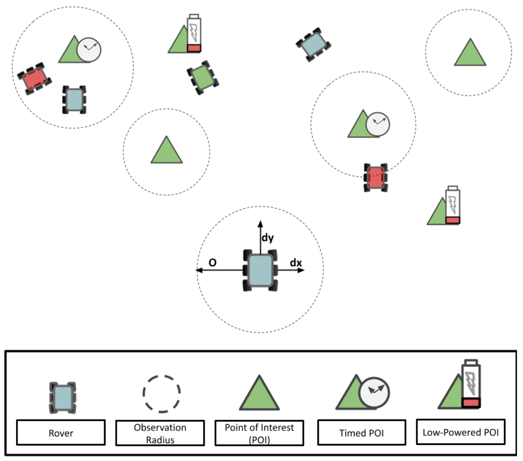

Figure 2: Various Points of Interest (POIs) must be jointly observed to get a reward. An agent in the rover environment navigates using the (dx, dy) actions and observes rovers and POIs within a chosen observation radius O. Optimal teams will try to balance observing Timed and Low-Power POIs that require higher speeds and observation radii respectively.

4 EXPERIMENTAL SETUP

We evaluate the performance of BEHT on several scenarios created using a variant of the multiagent heterogeneous rover exploration problem [9].

4.1 Compared Baselines

The primary metric of performance for this work is the team fitness. We also look at the variety of discovered policies over the course of learning since behavioral diversity is conducive to getting a higher fitness on the team objective. Unlike QD methods, the focus is not a full coverage of the behavior space, but rather only discovery of regions of the behavior space that contribute to the team performance.

We compare our method with three baselines, each serving as a state-of-the-art in a particular dimension of the problem: 1) Multiagent Evolutionary Reinforcement Learning (MERL), that provides a two-tier architecture of gradient based and evolutionary optimization to learn in tight coupled multiagent settings [14]; 2) Malthusian Reinforcement Learning, which enables discovery of synergies in agents operating on shared resources via managing population dynamics [18]; and 3) The intrinsic curiosity module, that allows exploration of diverse behaviors via dense pseudo-rewards based on an agents' prediction error in predicting its observations in the next time step [22]. Our method, BEHT, aims to combine the strengths of all three baselines by enabling coordination in tightly coupled settings via diversity search and implicit management of population dynamics via an evolutionary algorithm.

4.2 Rover Coordination Problem

In this section, we briefly describe the original rover exploration problem and a variation for heterogeneous agents used in this work.

A team of rovers in a continuous two-dimensional space must observe points of interest (POIs) that are spread over the environment. A POI is successfully observed when a required number of rovers are within its observation radius. We call this the coupling requirement of the POI. Thus, to make a successful observation, rovers equal to or greater than the coupling requirement of the POI must simultaneously observe it from within the POI's observation radius. The rovers are equipped with two density sensors to detect rovers and POIs, represented by equations 1 and 2.

$$S_{rover,q} = \sum_{j \in J_q} \frac{1}{d(i,j)} \tag{1}$$

In equation (1), q is the quadrant, d measures the Euclidean distance between the sensing rover i and another rover j; J_q is the set of all rovers in quadrant q, that are within the observation radius of the rover i.

$$S_{POI,q} = \sum_{k \in K_q} \frac{v_k}{d(i,k)} \tag{2}$$

In equation (2), d measures the Euclidean distance between the sensing rover i and the POI k; K_q is the set of POIs in the quadrant q, that are within the observation radius of the rover i. Defined with equations (1) and (2), a rover's observation is a fixed sized vector that remains unchanged regardless of the number of agents and POIs in the environment.

The system reward is computed using Equation 3.

$$G(z) = \sum_k \frac{\prod_i N_{(i,k)} V_k}{\frac{1}{n} \sum_j d(i,k)} \tag{3}$$

$G(z)$ is defined for z, the joint state-action of the rovers, V_k is the value of the POI k, $N_{i,k}$ is an indicator function that is true if the rover i is within the observation radius of the POI k and $d(i,k)$ is the Euclidean distance between rover i and POI k.

Real life robot exploration missions are often significantly richer than this setting in agent diversity, tasks and coordination (coupling) constraints. We model some of this diversity by introducing energy constraints and a variety of POI variants that call for diverse observation strategies:

(1) Vanilla POIs: The default POIs from the rover exploration task that reward agents with a specific value (equation 3) when observed simultaneously.
(2) Timed POIs: Mission critical POIs that dictate urgency. At every time step, the value of a Timed POI is reduced by one.
(3) Low-Power POIs: Either Vanilla or Timed POIs that are harder to detect. The probability of a Low-Power POI being encoded in the rover's input state (equation 2) is proportional to the observation radius of the rover. Agents with higher observation radius (and hence higher energy consumption) are more likely spot these.

At the beginning of every episode, every rover is given a fixed number of energy units e. The amount of energy units required at time step t, is governed by equation 4.

$$e_t = 0.5 o_t + 0.3 v_t \tag{4}$$

Where, o_t is the observation radius of the rover at time t and v_t is the velocity. The coefficients in the equation can be adjusted based on the specific requirements of the problem. Rovers can choose active velocity and observation radius as part of their actions. Higher speeds and radii require more energy and will reduce a rover's time of operation in the environment. Figure 2 shows a rover's view in the environment.

The observation space for the behavior policies consists of density estimates per quadrant defined by equation 2 for each POI variant. The rovers also have a density estimate per quadrant for sensing other rovers as described by equation 1. In addition to the density sensors, the current available energy e, represented by a real number, is also encoded in the observation vector. Thus, the size of the observation vector is $(4 * 3 + 4 + 1) = 17$. Policies are represented as neural networks that map the input state vector to a corresponding (dx, dy) action pair that lies in $[-2.0, 2.0]^2$ and an observation radius $o_t$in $[3, 10]$. At every time step, a policy takes an action (dx, dy, o_t) based on the current state that decides its movement in the (x, y) direction and the observation radius that will be used for computing the observation vector during the next time step. For all experiments presented in this work, the size of the world is $40x40$ units, and each episode lasts for 50 time steps.

$$r_{i,t} = \frac{v_k}{d(i,k)} \tag{5}$$

Equation (5) defines the dense reward $r_{i,t}$ for rover i at time t which is used to train the policies. The function $d()$ measures the Euclidean distance between the sensing rover i and the closest POI k with a value v_k. During the quality diversity phase of our algorithm, the policy population is trained using PPO on this local reward.

The trajectory of the policy that is fed to the dimensionality reduction method is a vector of $(o_t, v_t, d_{r,t}, d_{p,t})$ tuples for every time step t, where o_t is the observation radius used by the agent, v_t is the current speed, $d_{r,t}$ is the distance to the closest rover and $d_{p,t}$ is the distance to the closest POI. For the rover coordination problem, the tuple $(o_t, v_t, d_{r,t}, d_{p,t})$ captures the rover's POI strategy (which POIs to observe) and the rover's coordination strategy at time t. For an episode length of 50 steps, each policy trajectory is a vector of size $4 * 50 = 200$.

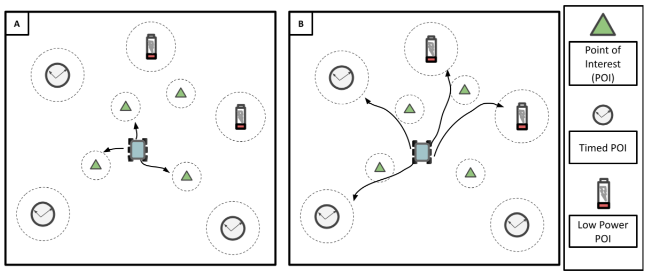

Figure 3: Experimental Setup 1: POIs are distributed radially. Higher valued Timed and Low-Power POIs are further away than Vanilla POIs. Agents start from the center.

5 RESULTS

Several experiments are conducted to quantitatively and qualitatively study the effectiveness of BEHT by observing the performance on the team-objective and discovered behaviors respectively. We use Principal Component Analysis (PCA) as our dimensionality reduction method since it has been used successfully in single-agent QD methods to learn the behavior space [6]. PCA is a widely used technique in machine learning and statistics that finds a linear projection of high dimensional data with linearly uncorrelated variables. In our experiments, PCA is used to project the high dimensional behavior trajectories into a low dimensional space which is used as the behavior space.

5.1 Multiple Independent Agents: Loose Coupling

Reinforcement learning methods are particularly good at evaluating the value of a given policy in an environment and further updating it to maximize a reward. However, in rich environments with large state and action spaces, finding good diverse policies would entail a computationally exhaustive search. The first experiment explores how systematically searching through the regions of the behavior space that yield rewards can be an helpful endeavor for finding diverse policies.

To make fair comparisons with single-agent training methods, we set the coupling count of the rover exploration problem to one: a single agent can visit a POI to get a reward equal to the value of the POI v_k. The three POI variants are distributed radially throughout the environment. The Vanilla POIs are closer to the center of the environment and are randomly assigned a reward value v_k between $[3, 5]$. The Timed and Low-Power POIs are distributed further from the center and have rewards in the range $[10, 15]$. Figure 3 shows this setup. There are 10 rovers (agents) and 12 POIs.

With limited time steps and energy units (given by equation 4), the local optimum in this setting is to visit the closest pois from the agent. Reinforcement learning methods are likely to get stuck in this optimum since visiting the higher valued outward POIs, requires an agent to take a long trajectory of sub-optimal actions (figure 3 (A)). In this loosely coupled scenario, because the agents do not require to coordinate their visits to the POIs, the optimal strategy is to travel outwards with higher velocity and large observation radius to visit the higher valued Timed and Low-Power POIs ((figure 3 (B)).

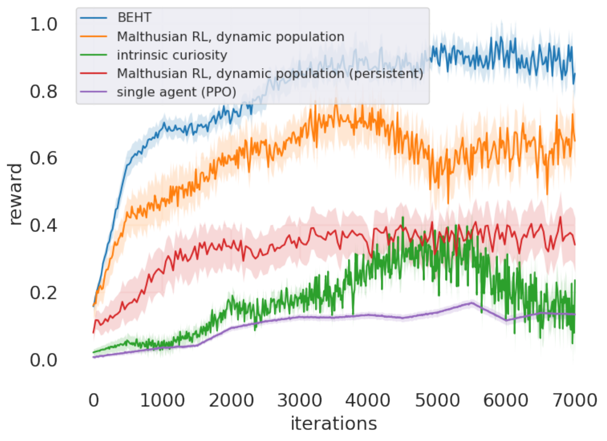

Figure 4: Performance as independent agents.

Figure 4 shows the normalized cumulative reward (summed values of visited POIs) obtained by BEHT and the baselines. The dense reward used for the QD-like iteration in BEHT is the Euclidean distance to the closest POI, given by Equation (5). As this is a single-agent setting, the team-objective used by the EA is the sum of rewards of all agents in the team. At every behavior space refinement step, PCA reduces the policy trajectories down to a three dimensional behavior space. Agents trained using BEHT learn to visit close to 90% of the POIs.

Malthusian Reinforcement Learning (MRL) relies on a suitable reward that forces agents to specialize. As agents can independently visit the same POIs and get rewarded, there is no implicit incentive in the reward signal to specialize. Agents using MRL therefore only visit the closest lower valued POIs. In figure 4, Malthusian RL (Persistent) shows the performance of agents when POIs are persistent. To allow MRL to perform optimally, we make the POIs non-persistent: a POI only gives a reward to the first agent that visits it. This allows agents with MRL to specialize in order to visit Timed and Low-Power POIs. Agents trained with BEHT perform slightly better than MRL since the specialization is not driven by the reward structure but exploration of the behavior space itself to maximize the reward. MRL (both plotted variants) is setup for four species ($L = 4$) and a dynamic population size.

Agents trained with the intrinsic curiosity module generate their own shaped rewards based on the prediction error in predicting the next observation. This allows agents to discover some of the POIs but they get stuck at the sub-optimal lower valued inner circle of Vanilla POIs. Occasionally, agents stumble upon the higher valued outer POIs but do not do so consistently.

5.2 Multiagent Coordination: Tight coupling

In this section, we explore how BEHT can be effective for learning strong synergies required for tight coupled coordination of heterogeneous agents. We perform tests on several coupling configurations to see how BEHT and the baselines can learn the required behavior diversity to reach the different POI variants and at the same time, favour behaviors that allow the team to coordinate the visits. There are 10 agents that start every episode from the center of the environment and 15 POIs (all variants) uniformly distributed throughout the environment. Unlike the previous experiment, since agents do not operate independently, POIs are not persistent. Once observed by a group of agents equal to or greater than the coupling requirement, the corresponding POI is removed from the environment.

BEHT uses the Euclidean reward (equation 5) for the QD-like phase and the global reward $G(z)$ (equation 3) as the fitness for the evolutionary algorithm phase. In all the tested coupling configurations (figure 5), agents using BEHT acquire sufficient diversity to observe all POI variants and learn to coordinate. As the coupling increases, the overall problem becomes significantly harder due

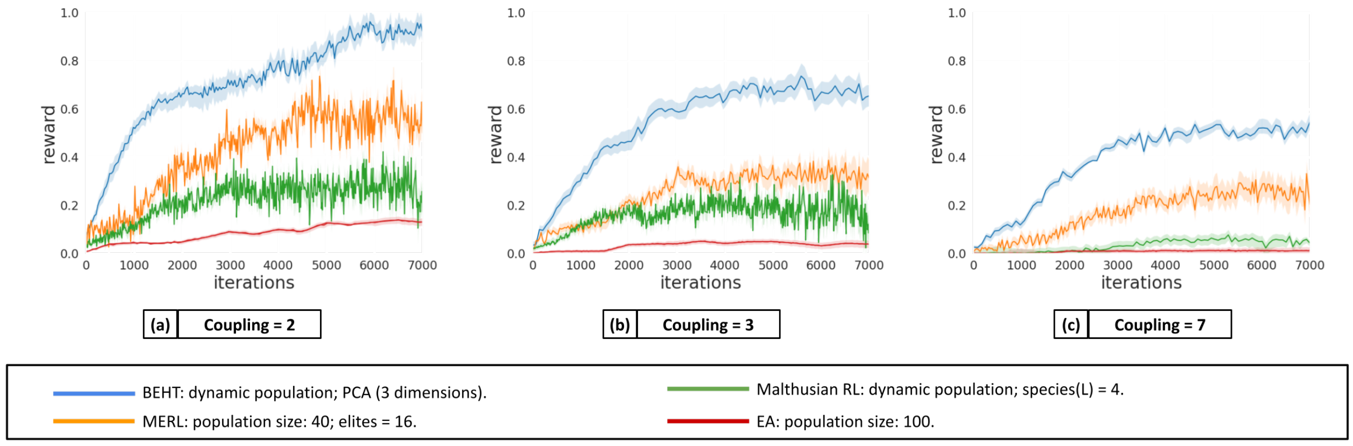

Figure 5: Performance on the tightly-coupled task for coupling requirement of 2, 3, and 7 respectively.

to increased reward sparsity and the overall team performance reduces, but stays significantly above the tested baselines.

MERL uses the same rewards (equations 5 and 3) as BEHT, but learns a sub-optimal team strategy. Although MERL has a comparable two-tier structure with dense agent-specific rewards and sparse team rewards, it fails to account for the needed diversity in agent behavior required to perform optimally. The degradation of performance with increased coupling is similar to BEHT as the evolutionary algorithm in MERL has to learn using an increasingly sparse team reward.

Teams trained with MRL are able to observe at most 40% of the total POIs for a coupling of two but the performance quickly drops to observing less than 10% POIs as the coupling is increased. The sub-optimal performance at lower coupling is expected since MRL does not explicitly encourage exploration of behavioral diversity. The evolutionary algorithm in MRL uses the team fitness to manage the population dynamics but does not apply selection pressure in the direction of the fitness. With sparser rewards, the population dynamics updates in MRL slow down due to the lack of gradient information which prevents new innovation in behavior as well as the incentive to coordinate.

5.3 Adaptation in a Dynamic Environment

To test how BEHT can systematically move through the behavior space in response to the changes in the environment, a dynamic environment is setup such that after every 2000 generations, the POIs in the environment change. In our test domain, this implies changing the distribution of the POI variants and their coupling requirements such that the currently stable behaviors are no longer optimal. This allows to inspect the effect of selection pressure from the evolutionary algorithm on discovering, refining and pruning behaviors. To understand the distribution of behavior policies in the behavior space, PCA projects the policy trajectories to a 2D behavior space.

The behaviors are clustered in the behavior space to allow qualitative analysis. We want to look at the distribution and relative population sizes of the stable behavior clusters after changes to the environment. Figure 6 shows the discovered behaviors that achieve the highest fitness over five consecutive epochs, each of 2000 generations (bottom row). On the top, the heat maps depict the distribution of POIs for each epoch and the potential trajectory of movement in the behavior space that might have led to the discovery of the clustered behaviors on the bottom row. Over the course as the distribution changes, new behaviors are discovered and older discarded or displaced (via mutation) in the direction necessary to maximize cooperative behavior.

In the first epoch the environment consists only of vanilla POIs. This is equivalent to the classic rover exploration problem, where behavioral diversity is largely unnecessary. Behaviors with higher velocities and observation radii are naturally more energy demanding (equation 4) and will have lower fitness (due to lower time of operation in the environment) and are therefore unfavourable. This is evident in the discovered stable behaviors (Epoch 1, bottom row).

In the second epoch, in addition to vanilla POIs, several Timed POIs are introduced. Rovers should prioritize high valued Timed POIs over vanilla POIs. This must be balanced with the energy requirement: If too many rovers learn behaviors for observing Timed POIs, the lower operation time due to increased velocity would imply that only a small percentage of the rovers will survive long enough in the episode to observe all the POIs, thus reducing the team fitness. As shown in figure 6, populations of three behavior clusters are stable in this epoch. Agents with behaviors from increased observation radii with slower speeds and reduced observation radii with faster speeds are driven towards discovering and observing Timed POIs. The third behavior cluster with average speed and observation radius is a carry-over from the last epoch that is still stable and enables observing vanilla POIs.

All POI variants are uniformly distributed in the third epoch leading to two dramatic changes. The behaviors with an average speed and observation radius has a significantly reduced population. At the same time, new behaviors with large observation radii and high speeds are discovered (bottom row). These can be critical for observing the increased number of Timed and Low-Power POIs. Potentially due to the higher energy requirement and consequent lower fitness, the number of such behaviors is very small.

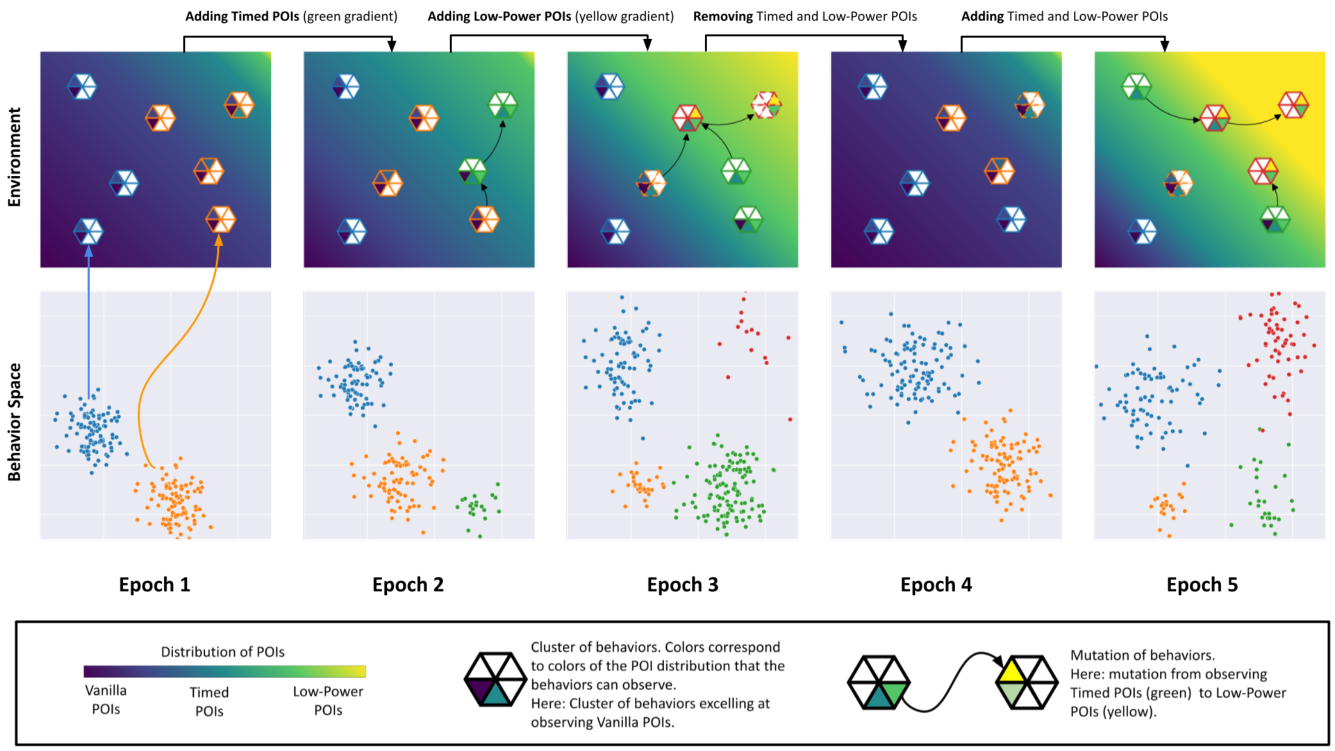
Adding Timed POIs (green gradient)
Adding Low-Power POIs (yellow gradient)
Removing Timed and Low-Power POIs
Adding Timed and Low-Power POIs
Environment
Behavior Space
Epoch 1
Epoch 2
Epoch 3
Epoch 4
Epoch 5
Distribution of POIs
Vanilla POIs
Timed POIs
Low-Power POIs
Cluster of behaviors. Colors correspond to colors of the POI distribution that the behaviors can observe.
Here: Cluster of behaviors excelling at observing Vanilla POIs.
Mutation of behaviors.
Here: mutation from observing Timed POIs (green) to Low-Power POIs (yellow).

Figure 6: The distribution of POIs in the environment and the potential diversity search trajectory (top row), and the corresponding optimal behaviors projected in the behavior space (bottom row) is shown for five consecutive epochs (1) through (5). The distribution of POIs is changed after each epoch. In the first and fourth epochs, the environment consists primarily of Vanilla POIs (dark blue gradient) which call for behaviors with average speed and small observation radius. The second epoch introduces Timed POIs (blue to green gradient) which must be prioritized. This allows discovery of high velocity behaviors (behaviors colored green in epoch two's behavior space). Epochs three and five have a uniform distribution of Vanilla, Timed and Low-Power POIs, requiring behaviors with varying speeds and observation radii to cooperate. Both epochs produce stable behaviors that specialize for each POI type (four behavior clusters in their corresponding behavior space). The change in relative populations of the clustered behavior populations in the five epochs, highlights the adaptive nature of BEHT.

The fourth epoch shows a collapse in behaviors that required higher energy as Timed POIs are removed from the environment. This regression in specialization is also favorable. In an exploratory mission with a fleet of specialized robots, identifying and limiting the capabilities needed to complete current goals can help to conserve valuable resources.

In the fifth epoch, all POI types are uniformly distributed. Stable behaviors resemble the behaviors seen in the third epoch closely, except for the change in population of different behaviors. This change roughly corresponds to the change in distribution of the POIs (top row).

Looking qualitatively at the discovered behaviors gives us further insights into the strength of BEHT. BEHT implicitly controls the population dynamics with the evolutionary algorithm and simultaneously filters the behavior space to discover promising behaviors.

6 CONCLUSION

We introduced BEHT, a multi-level training framework that allows systematic exploration of the agents' behavior space, required to complete diverse tasks as a coordinated team. BEHT decouples the search for behavioral diversity and fitness maximization. A Quality Diversity like approach uses gradient-based optimization to train agents to maximize dense agent-specific rewards and mutates them to fill the behavior space. Concurrently, an evolutionary algorithm maximizes the sparse team-based objective by applying selection, mutation and crossover operators to a population of agent teams. The selection operator essentially filters the successful regions of the behavior space while also balancing the number of agents of different behaviors.

The decoupling of the diversity search and optimization process allows the transformation of performance on the team-objective to direct the direction of functional diversity. The resulting method allows agents to progressively explore promising regions of the behavior space that promote team coordination on diverse goals.

In this work, BEHT used a fixed allocation of computational resources across the QD, EA and behavior space refinement phases. Exploring methods to dynamically allocate resources to the three phases and execute them concurrently is a potentially promising step for future work. Finally, we will explore how BEHT can be extended to more complex domains such as StarCraft [30], RoboCup [15] and general mixed cooperative-competitive settings.

REFERENCES

[1] Stefano V Albrecht and Peter Stone. 2018. Autonomous agents modelling other agents: A comprehensive survey and open problems. *Artificial Intelligence* 258 (2018), 66–95.

[2] Itamar Arel, Cong Liu, Tom Urbanik, and Airton G Kohls. 2010. Reinforcement learning-based multi-agent system for network traffic signal control. *IET Intelligent Transport Systems* 4, 2 (2010), 128–135.

[3] Jonathan C Brant and Kenneth O Stanley. 2020. Diversity preservation in minimal criterion coevolution through resource limitation. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*. 58–66.

[4] Lucian Bu, Robert Babu, Bart De Schutter, et al. 2008. A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 38, 2 (2008), 156–172.

[5] Cédric Colas, Vashisht Madhavan, Joost Huizinga, and Jeff Clune. 2020. Scaling map-elites to deep neuroevolution. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*. 67–75.

[6] Antoine Cully. 2019. Autonomous skill discovery with quality-diversity and unsupervised descriptors. In *Proceedings of the Genetic and Evolutionary Computation Conference*. 81–89.

[7] Felipe Leno Da Silva, Matthew E Taylor, and Anna Helena Reali Costa. 2018. Autonomously Reusing Knowledge in Multiagent Reinforcement Learning.. In *IJCAI*. 5487–5493.

[8] Sylvain Damiani, Gérard Verfaillie, and Marie-Claire Charmeau. 2005. An earth watching satellite constellation: How to manage a team of watching agents with limited communications. In *Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*. 455–462.

[9] Gaurav Dixit, Nicholas Zerbel, and Kagan Tumer. 2019. Dirichlet-Multinomial Counterfactual Rewards for Heterogeneous Multiagent Systems. In *2019 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*. IEEE, 209–215.

[10] Kurt Dresner and Peter Stone. 2005. Multiagent traffic management: An improved intersection control mechanism. In *Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*. 471–477.

[11] Jakob Foerster, Richard Y Chen, Maruan Al-Shedivat, Shimon Whiteson, Pieter Abbeel, and Igor Mordatch. 2018. Learning with opponent-learning awareness. In *Proceedings of the 17th International Conference on Autonomous Agents and Multi-Agent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 122–130.

[12] Adam Gaier, Alexander Asteroth, and Jean-Baptiste Mouret. 2020. Automating representation discovery with map-elites. *arXiv preprint arXiv:2003.04389* (2020).

[13] Jared Hill, James Archibald, Wynn Stirling, and Richard Frost. 2005. A multi-agent system architecture for distributed air traffic control. In *AIAA guidance, navigation, and control conference and exhibit*. 6049.

[14] Shauharda Khadka, Somdeb Majumdar, Santiago Miret, Stephen McAleer, and Kagan Tumer. 2019. Evolutionary reinforcement learning for sample-efficient multiagent coordination. *arXiv preprint arXiv:1906.07315* (2019).

[15] Hiroaki Kitano, Minoru Asada, Yasuo Kuniyoshi, Itsuki Noda, and Eiichi Osawa. 1997. Robocup: The robot world cup initiative. In *Proceedings of the first international conference on Autonomous agents*. 340–347.

[16] Guillaume J Laurent, Laëtitia Matignon, Le Fort-Piat, et al. 2011. The world of independent learners is not Markovian. *International Journal of Knowledge-based and Intelligent Engineering Systems* 15, 1 (2011), 55–64.

[17] Joel Lehman and Kenneth O Stanley. 2011. Evolving a diversity of virtual creatures through novelty search and local competition. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*. 211–218.

[18] Joel Z. Leibo, Julien Perolat, Edward Hughes, Steven Wheelwright, Adam H. Marblestone, Edgar Duéñez Guzmán, Peter Sunehag, Iain Dunning, and Thore Graepel. 2019. Malthusian Reinforcement Learning. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems* (Montreal QC, Canada) *(AAMAS '19)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 1099–1107.

[19] Laetitia Matignon, Guillaume J Laurent, and Nadine Le Fort-Piat. 2012. Independent reinforcement learners in cooperative markov games: a survey regarding coordination problems. *The Knowledge Engineering Review* 27, 1 (2012), 1–31.

[20] Jean-Baptiste Mouret and Jeff Clune. 2015. Illuminating search spaces by mapping elites. *arXiv preprint arXiv:1504.04909* (2015).

[21] Jørgen Nordmoen, Kai Olav Ellefsen, and Kyrre Glette. 2018. Combining MAP-elites and incremental evolution to generate gaits for a mammalian quadruped robot. In *International Conference on the Applications of Evolutionary Computation*. Springer, 719–733.

[22] Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. 2017. Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*. PMLR, 2778–2787.

[23] Justin K Pugh, Lisa B Soros, and Kenneth O Stanley. 2016. Quality diversity: A new frontier for evolutionary computation. *Frontiers in Robotics and AI* 3 (2016), 40.

[24] A. Rahmattalabi, J. J. Chung, M. Colby, and K. Tumer. 2016. D++: Structural credit assignment in tightly coupled multiagent domains. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2016)*. 4424–4429.

[25] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* (2017).

[26] Peter Stone, Gal Kaminka, Sarit Kraus, and Jeffrey Rosenschein. 2010. Ad hoc autonomous agent teams: Collaboration without pre-coordination. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 24.

[27] Ming Tan. 1993. Multi-Agent Reinforcement Learning: Independent vs. Cooperative Agents. In *In Proceedings of the Tenth International Conference on Machine Learning*. Morgan Kaufmann, 330–337.

[28] Claire Tomlin, George J Pappas, and Shankar Sastry. 1998. Conflict resolution for air traffic management: A study in multiagent hybrid systems. *IEEE Transactions on automatic control* 43, 4 (1998), 509–521.

[29] Karl Tuyls and Gerhard Weiss. 2012. Multiagent learning: Basics, challenges, and prospects. *Ai Magazine* 33, 3 (2012), 41–41.

[30] Oriol Vinyals, Timo Ewalds, Sergey Bartunov, Petko Georgiev, Alexander Sasha Vezhnevets, Michelle Yeo, Alireza Makhzani, Heinrich Küttler, John Agapiou, Julian Schrittwieser, et al. 2017. Starcraft ii: A new challenge for reinforcement learning. *arXiv preprint arXiv:1708.04782* (2017).

[31] Jijun Wang, Michael Lewis, and Paul Scerri. 2006. Cooperating robots for search and rescue. In *Proceedings of AAMAS Workshop on Agent Technology for Disaster Management*.