Learning Cooperation with Learned Skills

Jusuk Lee

***Abstract*—Multi-agent reinforcement learning(MARL) algorithms typically require the design of a sophisticated reward function to effectively guide multiple agents in learning desirable behaviors. However, this process is time-consuming and demands domain-knowledge. Can we achieve cooperative behavior and solve multi-agent tasks using only a simple sparse reward, such as giving a reward for success? In this paper, we train each agent learns a diverse set of skills in advance. Then the agents are able to solve complex tasks by effectively coordinating these skills. Experiments show that our method can solve complex robotic cooperation tasks using only sparse rewards.**

***Index Terms*—Deep learning methods, Unsupervised skill discovery, Sparse reward**

I. Introduction

MULTI-agent reinforcement learning(MARL) holds significant potential due to the inherently multi-agent nature of many real-world scenarios. For instance, domains such as unmanned aerial vehicles[1], autonomous driving[2] necessitate the interaction and coordination among multiple agents to achieve optimal performance. However, a common challenge in MARL is the need to engineer a complex reward function that requires domain-specific knowledge. As the number of agents increases, the environment becomes more complex and the behavior space expands. This complexity necessitates more sophisticated reward shaping to effectively solve multi-agent tasks. Learning to solve cooperative tasks from scratch requires a lot of interaction with and extensive reward engineering[3]. Additionally, the number of reward terms used as objective functions increases, leading to the necessity of adjusting the weights of these reward terms[4]. This process can be time-consuming and may inadvertently induce undesirable behaviors.

One promising method to solve sparse reward multi-agent problems is to first enable agents to learn a diverse set of skills and then effectively coordinate them to accomplish tasks. For example, consider a basketball game, where players first learn individual skills such as shooting, passing, and dribbling. Then, by appropriately combining and coordinating these skills, players can successfully score points and win the game. This approach involves decomposing complex tasks into simpler, manageable skills, which can be learned independently and then integrated to achieve the overall objective.

Learning diverse skills without reward has been studied in multi-agent scenarios[8]-[10]. Existing methods for multi-agent scenarios have predominantly used mutual information (MI)-based approaches. However, due to the limitations of MI-based methods, they fail to learn diverse behaviors with a smaller state space coverage[11]-[12]. Therefore, they have not been applied to complex environments involving robots, and have only been used in simpler environments such as video game scenarios[8]-[10]. Additionally, there is a study that applied in robot environments using multiple sub-skills[4]. However, they also utilize MI-based method to learn sub-skills. To compensate for the small state space coverage of MI-based methods, they used a complex dense reward function.

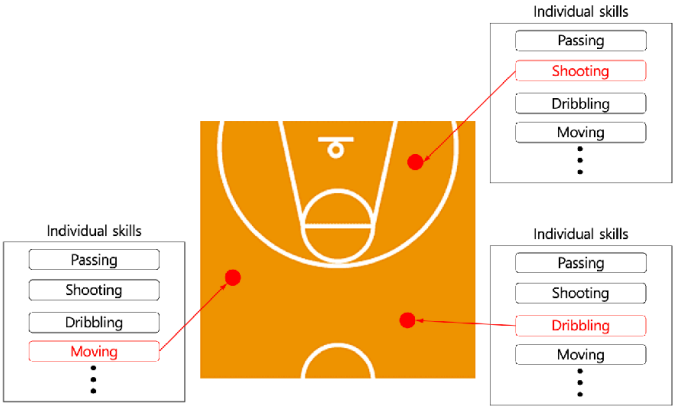


Fig. 1: Illustration of individual skills in basketball.

Our paper addresses sparse reward multi-agent problem by decomposing complex cooperative behaviors into simple, manageable behavior. Subsequently, we combined these simple behaviors to solve multi-agent complex tasks. Unlike traditional MI-based methods, we used a metric-based skill discovery method[13] to obtain primitive behaviors. Metric-based methods incorporate the concept of a metric, allowing an agent to quantify how different each state is. This enables the learning of a much more diverse set of primitive behaviors(i.e. skills). To avoid designing a complex reward function, we utilized Hindsight Experience Replay(HER)[14] to train meta policy. The meta policy coordinated the skills of each agent appropriately, thereby solving complex cooperative tasks.

Our main contribution is an algorithm to solve multi-agent tasks with only sparse reward in complex environments such as robot. The key idea of the algorithm is to decompose the complex behaviors into simple, diverse behaviors, then coordinate these behaviors properly. We created three cooperative tasks(Two-ant reach, Two-ant bar push, Two-manipulator bar push). And we demonstrated empirically that our method can solve many complex multi-robot tasks.

II. Related Work

Learning to solve multi-agent task either sparse reward or considerable engineered reward is complex[4]. In this section, we will discuss related prior methods to solve multi-agent task with sparse reward.

This paper was produced by the IEEE Publication Technology Group. They are in Piscataway, NJ.

Manuscript received April 19, 2021; revised August 16, 2021.

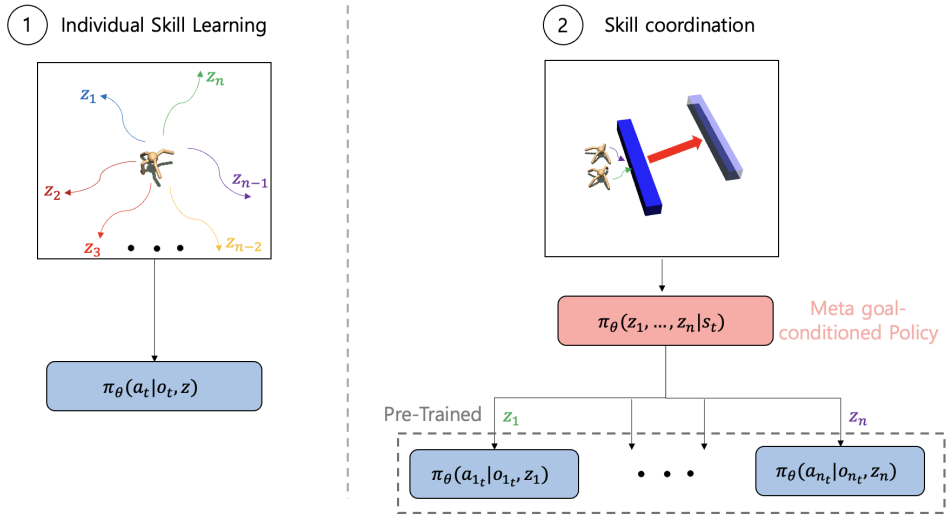


Fig. 2: Our approach

Firstly, **MARL** algorithms have been developed to address the sparse reward issue. SEAC[15] employs experience sharing among agents to facilitate efficient exploration. CMAE[16] enhances cooperative exploration by selecting a common goal for agents from the state space. VACL[17] addresses the sparse reward problem by using an automatic curriculum learning algorithm that gradually increases the complexity of training tasks. More recently, MASER[18] has introduced a goal-conditioned approach that generates subgoals for agents using the experience replay buffer. However, these approaches suffer from the credit assignment problem. Moreover, as the number of agents increases, the joint behavior space and state space to be explored becomes larger. This makes it time-consuming and highly challenging to train a multi-agent system from scratch[3]. In this work, we propose a method where each agent first learns diverse and distinguishable skills. Then, by coordinating these skills, we aim to effectively solve multi-agent tasks. Our approach involves breaking down complex tasks into simpler components and then combining these simpler skills to accomplish more complex tasks.

Secondly, **Mutual information-based skill discovery** is promising method for enabling agents to learn diverse behaviors without any rewards[5]. This approach maximizes the mutual information (MI) between states S and skills Z, denoted as $I(S, Z)$. By doing so, it establishes a correlation between the skill latent vector z and the states S, resulting in distinguishable state trajectories for different skill z values. MASD[10] utilizes a shared team skill Z. It maximizes the mutual information between the global state X and the skill Z, $I(f(X); Z$. While it minimizes the mutual information between each agent's observation $X^{(i)}$ and the skill Z, $I(f(X^{(i)}; Z)$. This ensures that the shared skill Z is related to the combination of all states rather than being associated with any single agent. However, this method suffers from high complexity as the number of agents increases. HMASD[8] simultaneously learns team skill Z and individual skills $z_{(i)}$, enabling agents to acquire a diverse set of skills tailored to the task requirements. This approach effectively addresses sparse multi-agent tasks by leveraging both types of skills. However, both of these methods have only been applied to simple environments such as multi-particle[21] and Overcooked[22], rather than complex environments like robots. This limitation is due to the constraints of the mutual information approach. MI objective does not ensure coverage of a larger region in the state space[11]. This is because MI remains invariant under traveled distances or any invertible transformation, meaning $I(S; Z) = I(f(S); Z)$ for any invertible f[12]. As seen in Equation (1), MI objective is defined by KL divergence, which is a metric-agnostic quantity. As a result, it focuses only on the distinguishability of behaviors without considering the magnitude of their differences. Consequently, it ends up to discover only simple, static skills.

$$I(S; Z) = D_{KL}(p(s, z)||p(s)p(z)) \tag{1}$$

Our most closely related work is Modular-SBD[4]. Similar to our approach, they introduced a hierarchical structure in a robotic environment. Initially, each agent learns primitive skills, which are then combined to solve multi-agent tasks. To learn primitive skills, they combined the MI objective and complex dense reward to encourage exploration and acquire more diverse behaviors, i.e. dynamic skills. However, we employ a metric-aware quantity that allows agents to learn dynamic skills wihtout the need for additional dense reward shaping. Furthermore, while they used separate reward for skill coordination, we leverages goal-conditioned reinforcement learning to solve tasks using only sparse reward.

III. PRELIMINARIES

A. Metric-based Skill Discovery

Metric-based skill discovery have been proposed to resolve the limitation of MI-based skill discovery. It is also unsupervised reinforcement learning to learn diverse set of behaviors without reward and human prior. LSD[11], CSD[12], METRA[13] are notable examples of this approach. It aims to not only maximize the relationship between state S and skill

Z but also maximize a specific metric quantity. For instance, LSD[11] used the Euclidean distance traveled in the state space for each skill, CSD[12] employes a controllability-aware distance, and METRA[13] relies on temporal distance. Unlike mutual information-based skill discovery, metric-based skill discovery incorporates the concept of a metric. This allows agents to understand how different each skill is, leading to much more diverse set of behaviors.

B. Hindsight Experience Replay

Hindsight Experience Replay (HER)[14] is a technique within goal-conditioned reinforcement learning (GCRL) that enhances sample efficiency and addresses the challenges of sparse rewards. In traditional reinforcement learning, agents often struggle to learn from sparse rewards due to the rarity of receiving positive feedback. HER mitigates this issue by allowing agents to learn from all experiences, regardless of whether the original goal was achieved. HER works by replaying past experiences with a modified goal. Specifically, if an agent fails to achieve its intended goal, the experience is replayed with the goal replaced by the state that was actually reached. This way, the agent learns as if the achieved state was the desired goal. For example, in a robotic manipulation task where the goal is to move an object to a target location, if the robot accidentally moves the object to a different location, HER treats this new location as the goal during replay, providing valuable learning signals. The primary benefit of HER is its ability to facilitate learning from sparse rewards. By using achieved goals as substitutes, HER ensures that every episode provides useful learning experiences, reducing the need for intricate reward shaping. This approach significantly improves sample efficiency and accelerates the learning process.

IV. METHODS

A. Training primitive skills

We utilize METRA[13] to train an agent to learn diverse behaviors without reward and human prior knowledge. METRA[13] trains an agent to not only establish mapping between S and Z but also maximize the temporal distance in latent space. Consequently, the agent can learn skills that are distinguishable and temporally distant from each other. Diverse behaviors conditioned on a skill latent vector z can be achieved by maximizing the Wasserstein dependency measure(WDM) between states and skills. Unlike the KL divergence used in MI-based methods, WDM have a metric. This allows it to use the metric to measure the temporal distance between states, enabling the learning of dynamic skills. However, METRA[13] does not take into account specific interactions with objects. In this work, we also incorporated a mutual information term to ensure that the agent learns skills related to interactions with specific objects in manipulation environments. The objective can be written as follows:

$$I_W(S;Z) + \alpha I(S;Z) \tag{2}$$

$$= W(p(s,z), p(s)p(z)) + \alpha D_{KL}(p(s,z), p(s)p(z)) \tag{3}$$

where $I_W(S;Z)$ is the Wasserstein dependency measure, and W is the 1-Wasserstein distance on the metric space $(S\mathrm{x}Z, d)$ with distance metric d. This object yield the following objective:

$$\begin{aligned} &\sup_{\pi,\phi} E_{p(\tau,z)}\left[\sum_{t=0}^{T-1}(\phi(s_{t+1}) - \phi(s_t))^T z\right] \\ &\quad + \alpha\left(H(a|s,z) + E_{z\sim p(z), s\sim\pi(z)}\left[\log q_\psi(z|s) - \log p(z)\right]\right) \\ &\text{s.t. } \|\phi(s) - \phi(s')\|_2 \le 1,\ \forall (s,s') \in S_{\text{adj}} \end{aligned} \tag{4}$$

where ϕ: $S \to R^D$ is mapping function from state space to latent space, discriminator $q_\psi(z|s)$, and S_{adj} denotes the set of adjacent state pairs in the MDP. We recommend the readers to METRA[13] and DIAYN[5] for more detail derivation. During training, the agent learns a primitive behavior policy $\pi(a|s,z)$ dependent on a random skill vector z. Further details of the training scheme are provided in Algorithm 1.

Algorithm 1 First step, train behavior primitives

Initialize skill policy $\pi(a|s,z)$, representation function $\phi(s)$, discriminator $q_\psi(z|s_{t+1})$, Lagrange multiplier λ, replay buffer D
for i $\leftarrow$ 1 to (N epochs) **do**
  for j $\leftarrow$ 1 to (M episodes) **do**
    Sample skill $z \sim p(z)$
    Sample transitions (s_t, a_t, s_{t+1}) with $\pi(a|s,z)$ and store in replay buffer D
  end for
  for k $\leftarrow$ 1 to (K updates) **do**
    Update discriminator (ψ) with SGD.
    Update $\phi(s)$ to maximize
    $E_{s,z,s'\sim D}[(\phi(s') - \phi(s))]^T z$
    $+\lambda min(\epsilon, 1 - ||\phi(s') - \phi(s)||_2^2)$
    Update λ to minimize
    $E_{s,z,s'\sim D}[\lambda min(\epsilon, 1 - ||\phi(s') - \phi(s)||_2^2]$
    Set skill reward
    $r = (\phi(s') - \phi(s))^T z + \alpha(log q_\psi(z|s_{t+1}) - log p(z))$
    Update policy $\pi(a|s,z)$ to maximize r with SAC.
  end for
end for

B. Coordination skills among multiple agents

Using the primitive behavior policy learned in the first step, we train a meta policy $\pi_{meta}(z_1, ..., z_n|s)$. The meta policy chooses appropriate primitive skills for each agent to solve multi-agent tasks. Once the skills are selected by the meta policy, each agent takes action based on its current observation and the assigned skill(i.e. $\pi_i(a_{t_i}|o_{t_i}, z_i)$). The high-level meta policy and the low-level primitive behavior policy operate on different time horizons. Additionally, to address sparse reward settings, this work trains the meta policy using Soft Actor-Critic(SAC)[19] and Hindsight Experience Replay(HER)[14]. More details of training to coordinating phase are provided in Algorithm 2.

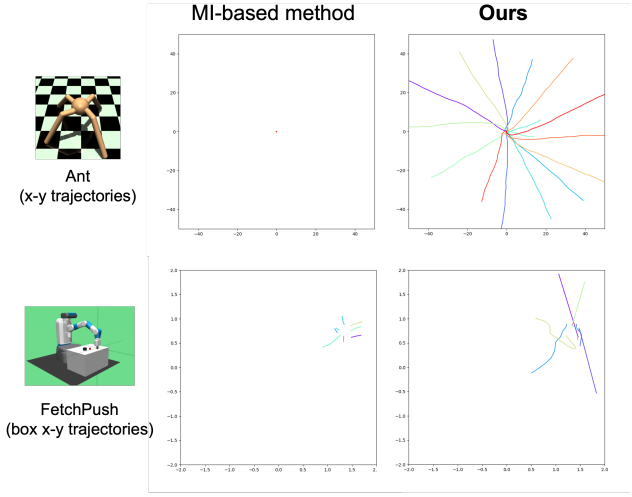


Fig. 3: **Examples of 16 behaviors learned by MI-based method and ours.** For ant environment, we plot the x-y trajectories of ant robot. For FetchPush environment, we plot the x-y trajectories of box.

Algorithm 2 Second step, train skill coordination.

Initialize meta policy $\pi_{meta}(z_1, ..., z_n|s)$, sets of pretrained low-level policies $\pi_1(a_1|o_1, z_1), ..., \pi_n(a_n|o_n, z_n)$, replay buffer R, meta policy frequency d
for t $\leftarrow$ 1 to (M episodes) **do**
  Sample a goal g.
  if t mod d **then**
    Sample an skills $z_1, ..., z_n$ using meta policy $\pi_{meta}(z_1, ..., z_n|s_t, g)$
  end if
  Sample an action $a_{t_1}, ..., a_{t_n}$ from pretrained low-level policy $\pi_1(a_1|o_1, z_1), ..., \pi_n(a_n|o_n, z_n)$
  Execute $A = a_{t_1} \cup ... \cup a_{t_n}$ and observe a new state s_{t+1}
  if t mod d **then**
    $r_t = (s_t, (z_{t_1}, ..., z_{t_n}), g)$
    Store transitions $(s_t||g, (z_{t_1}, ..., z_{t_n}), r_t, s_{t+1}||g)$ in replay buffer R.
    Sample a set of additional goals for replay
    for $g' \in G$ **do**
      $r' = r(s_t, (z_{t_1}, ..., z_{t_n}), g')$
      Store transitions $(s_t||g', (z_{t_1}, ..., z_{t_n}), r_t, s_{t+1}||g')$ in R.
    end for
    for k $\leftarrow$ 1 to (K updates) **do**
      Update policy $\pi_{meta}(z_1, ..., z_n|s)$ with SAC.
    end for
  end if
end for

V. EXPERIMENTS

We present empirical evidence to demonstrate that our method can solve sparse multi-agent tasks. To do so, we first show that the primitive behavior policy learns more dynamic skills compared to traditional MI-based methods. Then, we demonstrate how these skills are utilized to solve problems in a multi-agent task environments.

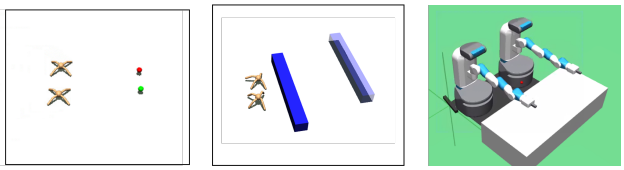


Fig. 4: Three tasks of manipulation and locomotion. (a) Two ant robots are required to reach goals. (b) Two ant robots are required to push the block to the goal position. (c) Two fetch manipulator are required to push the block to the goal position.

A. How many diverse behaviors were learned

we trained an agent in the FetchPush manipulator environment and the Ant locomotion environment to develop a diverse range of behaviors. The agent was trained to maximize the objective described in Equation (4). We compared our method to traditional MI-based methods, specifically DIAYN, demonstrating through Fig. 3 that our approach achieves broader state coverage and learns more dynamic skills.

B. Muti-agent task benchmark

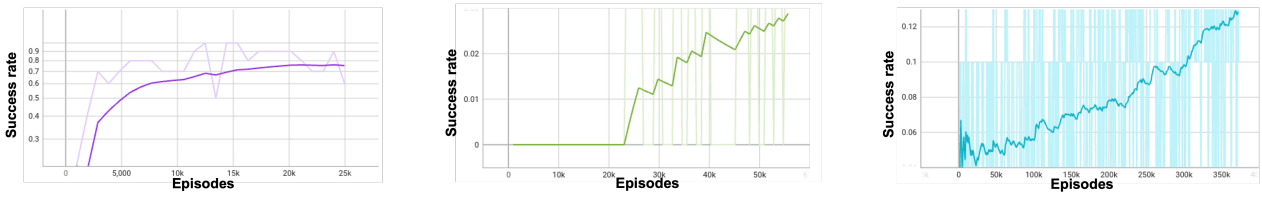
As shown in Fig. 4, we created a total of three tasks. The first task involves two ant robots cooperating to reach two different goals without predetermined assignments. The second task requires two ant robots to work together to push a long bar to the goal location. The third task is similar to the second, but invovles two manipulators cooperating to push a long bar to the goal location. The results of these tasks are shown in Fig. 5. While the first task was solved, the second and third tasks were not successfully learned, despite the successful learning of diverse primitive skills. The reasons for the lack of learning will be discussed in the next section.

VI. DISCUSSION

We speculate three main reasons for the failure of the meta policy to learn despite the successful learning of primitive skills.

First, **the stochastic properties of the custom environment** introduce variability that complicates learning process. For example, if the box is too light, the ant robot may push it with the same primitive skill, causing the box to fly away. Conversely, if the box is too heavy, the ant robot fail to push it and instead flip over. As this is a custom environment, it is necessary to adjust the weight and friction of the box to suit the ant robot through trial and error, and this process may have been insufficient.

Second, there is **a lack of precise interaction in the learned primitive skills**. We combined METRA[13] and mutual information-based methods to learn primitive skills. METRA[13] allows the agent to learn dynamics skills that



(a) (b) (c)

Fig. 5: Results of three benchmarks. (a) Two ant robots are required to reach goals (b) Two ant robots are required to push the block to the goal position (c) Two fetch manipulator are required to push the block to the goal position..

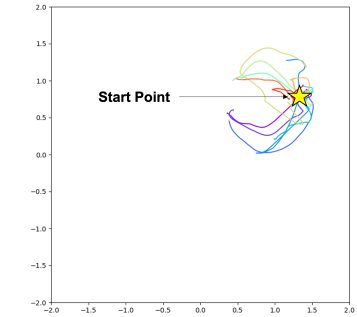
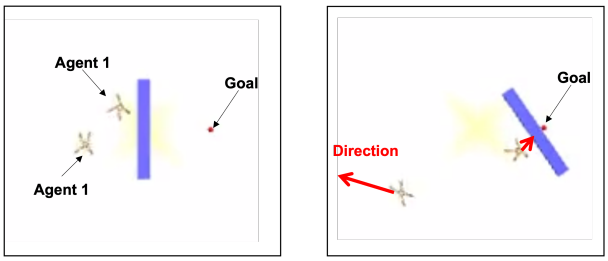


Fig. 6: 16 primitive skills learned by ours method in FetchPush environment. We plot the x-y trajectories of end effector. As you can see, the learned skills aim to diversify as far from the starting point as possible.



(a) (b)

Fig. 7: The case without demonstrating cooperation. (a) Initialization. (b) Success.

can cover a larger state space coverage. However, because METRA[13] uses temporal distance as a metric, it tends to learn skills that maximize temporal separation. As a result, the agent aims to learn skills that diverge as far as possible over time, causing the agent to move at maximum speed. As seen in Fig. 6, most of the skills tend to move as far away from the starting point as possible. From the meta policy's perspective, it can only direct the agent's actions in a specific direction and cannot influence the speed of the learned primitive skills. Therefore, this approach is not suitable for tasks like block pushing, where the agent needs to adjust its speed when in contact with an object.

Third, **the limitation of the HER algorithm** may have impacted the ability to learn the tasks successfully. HER[14] is a type of goal-conditioned RL. In environments like the Push task, the goal is specified only for the box, meaning the task is considered successful if the box reaches the goal. As a result, cooperative behaviors where two ant robot push the box together were not effectively developed. As seen in Fig. 7, the task can be solved without showing cooperative behavior.

VII. FUTURE WORK

Future work will be proceeded with addressing the three issues identified above. First, instead of creating a custom environment, we plan to utilize already validated environments. Specifically, we intend to use the environments employed in Modular-SBD[4], such as Jaco pick-push-place, Jaco bar-moving. Second, we plan to use the CSD[12] method, which enables the agent to learn skills that involve more interaction with objects compared to METRA[13]. Third, instead of using SAC+HER for the meta policy, we have a plan to utilize example-based control using success example[20]. This method can encourage cooperative behavior, using success examples that tasks are solved through cooperation.

REFERENCES

[1] D. Xu and G. Chen, "Autonomous and cooperative control of UAV cluster with multi-agent reinforcement learning," *The Aeronautical Journal*, vol. 126, no. 1300, pp. 932-951, 2022.
[2] B. Ravi Kiran *et al.*, "Deep reinforcement learning for autonomous driving: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 6, pp. 4909-4926, 2021.
[3] M. Riedmiller *et al.*, "Learning by playing solving sparse reward tasks from scratch," in *Proc. Int. Conf. Machine Learning*, 2018, pp.
[4] Y. Lee, J. Yang, and J. J. Lim, "Learning to coordinate manipulation skills via skill behavior diversification," in *Proc. Int. Conf. Learning Representations*, 2019.
[5] B. Eysenbach *et al.*, "Diversity is all you need: Learning skills without a reward function," *arXiv preprint arXiv:1802.06070*, 2018.
[6] A. Sharma *et al.*, "Dynamics-aware unsupervised discovery of skills," *arXiv preprint arXiv:1907.01657*, 2019.
[7] D. Cho, J. Kim, and H. J. Kim, "Unsupervised reinforcement learning for transferable manipulation skill discovery," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 7455-7462, 2022.
[8] M. Yang *et al.*, "Hierarchical Multi-Agent Skill Discovery," in *Advances in Neural Information Processing Systems*, vol. 36, 2024.
[9] J. Yang, I. Borovikov, and H. Zha, "Hierarchical cooperative multi-agent reinforcement learning with skill discovery," *arXiv preprint arXiv:1912.03558*, 2019.
[10] S. He, J. Shao, and X. Ji, "Skill discovery of coordination in multi-agent reinforcement learning," *arXiv preprint arXiv:2006.04021*, 2020.
[11] S. Park *et al.*, "Lipschitz-constrained unsupervised skill discovery," in *Proc. Int. Conf. Learning Representations*, 2021.

[12] S. Park *et al.*, "Controllability-aware unsupervised skill discovery," *arXiv preprint arXiv:2302.05103*, 2023.
[13] S. Park, O. Rybkin, and S. Levine, "Metra: Scalable unsupervised RL with metric-aware abstraction," *arXiv preprint arXiv:2310.08887*, 2023.
[14] M. Andrychowicz *et al.*, "Hindsight experience replay," in *Advances in Neural Information Processing Systems*, vol. 30, 2017.
[15] F. Christianos, L. Schäfer, and S. Albrecht, "Shared experience actor-critic for multi-agent reinforcement learning," in *Advances in Neural Information Processing Systems*, vol. 33, pp. 10707-10717, 2020.
[16] I.-J. Liu *et al.*, "Cooperative exploration for multi-agent deep reinforcement learning," in *Proc. Int. Conf. Machine Learning*, PMLR, 2021.
[17] J. Chen *et al.*, "Variational automatic curriculum learning for sparse-reward cooperative multi-agent problems," in *Advances in Neural Information Processing Systems*, vol. 34, pp. 9681-9693, 2021.
[18] J. Jeon *et al.*, "Maser: Multi-agent reinforcement learning with subgoals generated from experience replay buffer," in *Proc. Int. Conf. Machine Learning*, PMLR, 2022.
[19] T. Haarnoja *et al.*, "Soft actor-critic algorithms and applications," *arXiv preprint arXiv:1812.05905*, 2018.
[20] B. Eysenbach, S. Levine, and R. R. Salakhutdinov, "Replacing rewards with examples: Example-based policy search via recursive classification," in *Advances in Neural Information Processing Systems*, vol. 34, pp. 11541-11552, 2021.
[21] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in Advances in Neural Information Processing Systems 30, 2017.
[22] M. Carroll, R. Shah, M. Ho, T. Griffiths, S. Seshia, P. Abbeel, and A. Dragan, "On the utility of learning about humans for human-AI coordination," in Advances in Neural Information Processing Systems 32, 2019.