Multi-Agent Reinforcement Learning In Predator-Prey System

Candidate Number: 20011398, 23117025, 23048641

Abstract

This research paper explores how Multi-Agent Reinforcement Learning (MARL) can be used in a simulated predator-prey environment to develop intelligent behaviors in both collaborative and competitive settings. Our investigation hopes to provide an alternative approach for biologists to analyse complex interactions that are challenging to replicate in real-world experiments. We compare various MARL strategies, including decentralised and centralised training approaches utilising Q-learning and actor-critic methods. Furthermore, we devised distinct reward-shaping techniques and implemented ensemble methods to enhance the learning outcomes. Our findings indicate a notable enhancement in agent performance and demonstrate the differential impacts of selected training strategies on agent intelligence and robustness.

1 Introduction

The study of interaction dynamics between multiple agents in a shared environment forms a central aspect of reinforcement learning (RL), particularly in the domain of multi-agent reinforcement learning (MARL). One of the most extensively studied frameworks within this field is the prey-predator system. This system offers fundamental insights into competitive and cooperative interactions, as well as significant applications, including ecological modelling [3] and robotic swarm behavior[1].

The predator-prey system functions within a mixed cooperative-competitive environment. The competitive dynamic between the pursuing predator and the evading prey forces the agents adapt, learn and refine their strategies in response to the actions of other agents. Furthermore, effective strategies often incorporate elements of cooperation among either the prey or the predators. Therefore, the research questions listed below aims to enhance performance in both collaborative and competitive settings:

RQ 1. How does using various methods to encourage collaborative predator behavior influence the predator-prey system, and whether these methods can enhance the rewards for predators?
Motivation: By focusing on strategies that enhance rewards for predators, this research can lead to the development of more collaborative strategies. This not only improves outcomes in simulated environments but also offers insights that can be generalized to other areas like robotics or automated systems where collaboration is necessary.

RQ 2. How does using various methods to encourage collaborative prey behavior influence the predator-prey system, and whether these methods can enhance the rewards for prey?
Motivation: Investigating how different behaviors and strategies can enhance rewards for prey provides valuable information on how vulnerable agents can better protect themselves in a system. This not only makes the simulation more realistic but also helps in studying scenarios where less dominant agents interact with dominant ones. Moreover, prey cooperation is a relatively underexplored area contrasting with the more commonly researched predator collaborations.

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

RQ 3. How to prevent agents from overfitting the behaviour of their competitors, and what strategies can increase the robustness of agents in more unpredictable settings?
Motivation: Overfitting can severely limit an agent's ability to function effectively in new and unforeseen situations, thus limiting its overall utility. It is therefore necessary to address this problem.

Our contributions are:

1. We conducted extensive experiments using both decentralized (DQN and DDPG) and centralized (MADDPG) training strategies within the simulated predator-prey system. Our analysis focused on evaluating the impacts of these varied training approaches on the behavior and effectiveness of the agents involved.
2. We devised and applied distinct reward-shaping strategies for both predators and prey to promote cooperative behaviors among them. This approach has demonstrated an improvement in performance.
3. We implemented an ensemble training method that involves training a diverse set of K sub-policies. This approach has proven to enhance the robustness of agents, making them better handle changes in the policies of competing agents.

Code available at GitHub. [1]

2 Literature Review

In recent years, reinforcement learning (RL) has emerged as a powerful method for training agents to make decisions in complex environments. Different algorithms have been proposed to enable agents to learn from interactions and improve their performance over time. Q-learning is a classic algorithm [9] which provides a framework for agents to learn optimal policies in a discrete action space environment through the update of a Q-table. It can be applied to solve multi-agent problems where the state and action spaces are well-defined. However, its application in multi-agent systems presents unique challenges such as non-stationarity of the environment and exponential growth in the state-action space as the number of agents increases.

Deep Q-Networks (DQN) [7] extended the capabilities of classical Q-learning by integrating deep neural networks to approximate the Q-value functions, addressing the scalability issues associated with large state spaces. Though it shows its potential in multi-agent environments, its application is also hindered by issues such as overestimation of Q-values and instability due to the non-stationary policy of other agents. Lillicrap et al. [5] proposed a Deep Deterministic Policy Gradient (DDPG), which merges ideas from DQN and policy gradient methods to handle continuous action spaces. It utilizes a deterministic policy network to select actions and a Q-network to evaluate the action's utility, incorporating techniques such as experience replay and target networks to stabilize learning.

Despite the advances provided by these methodologies, their adaptation to multi-agent contexts is not straightforward. The above methods are all decentralized learning, forcing agents to train separately. To enable agents to have access to the observations and actions of all the other agents, MADDPG[6] was introduced and it maintains a central critic that learns from the aggregated information across all agents, while each agent operates based on its own local observations during execution. The emergence of MADDPG has paved the way for our research into the mechanisms of cooperation among agents, especially in predator-prey systems.

In addition, group hunting has been shown as an effective strategy in multi-agent environments[9] by introducing explicit cooperative mechanisms among predators. The dynamics of group hunting or group defense offer a rich context for exploring cooperative strategies within multi-agent systems. To stimulate group behavior, some techniques such as reward shaping can be particularly effective by allowing designers to encourage specific behaviors that promote cooperation or competition among agents. Previous research [2] has demonstrated that carefully crafted rewards can significantly influence agent behaviors, guiding them towards more efficient and effective strategies, and possibly accelerate the learning process. Lastly, to better simulate real-world environments with high degrees of uncertainty and variability, there are ensemble methods, which involve training multiple policies and randomly selecting them to make decisions. It has been shown [6] to effectively address challenges

[1] Code available at: `https://github.com/Julie3399/MARL_Predator_Prey`

like overfitting to specific opponent strategies, thereby enabling agents to perform well across a broader range of scenarios.

3 Background

3.1 Decentralized Training

3.1.1 Q learning and Deep Q-Networks(DQN) and Deep Deterministic Policy Gradient (DDPG)

The formula for Q-learning [9] is shown below:

$$Q^{new}(S_t, A_t) \leftarrow (1-\alpha) \cdot Q(S_t, A_t) + \alpha \cdot (R_{t+1} + \gamma \cdot \max_a Q(S_{t+1}, a)) \tag{1}$$

where α is the learning rate such that $0 < \alpha < 0$, $Q(S_t, A_t)$ is the current Q value, R_{t+1} is the reward received for the agent moving from state S_t to state S_{t+1}, γ is the discount factor, $\max_a Q(S_{t+1}, a))$ is the estimate of optimal future reward value from state S_{t+1}.
DQN [8] uses a neural network function approximator with weights θ as a Q-network to be more practical and have generality ability. Q network is trained by minimising a sequence of loss functions Li(i)$L_i(\theta_i)$ that changes at each iteration i.

$$L_i = \mathbb{E}_{s,a\sim\rho(\cdot)}[(y_i - Q(s, a; \theta_i)^2] \tag{2}$$

where $y_i = \mathbb{E}_{s'\sim\mathcal{E}}[r + \gamma \max_{a'} Q(s', a'; \theta_{i-1})|s, a]$ is the target for iteration i and $\rho(s, a)$ is a probability distribution over sequences s and actions a as behaviour distribution, $\mathcal{E}$ is the environment. By fixing the parameters from the last iteration, the gradient is computed to minimise the loss by the following formula:

$$\nabla_{\theta_i} L_i(\theta_i) = \mathbb{E}_{s,a\sim\rho(\cdot);s'\sim\mathcal{E}} \left[\left(r + \gamma \max_{a'} Q(s', a'; \theta_{i-1}) - Q(s, a; \theta_i) \right) \nabla_{\theta_i} Q(s, a; \theta_i) \right]. \tag{3}$$

The Deep Deterministic Policy Gradient (DDPG) is used to overcome the limitation of Q-learning (does not work well in the large continuous state space) and the limitation of DQN(instabilities and inefficiency issues in the high-dimensional action/state space). DDPG extends the policy gradient framework to deterministic policies $\mu_\theta : S \rightarrow A$ and the gradient of the objective $J(\theta) = \mathbb{E}_{s\sim p^\mu}[R(s, a)]$ is

$$\nabla_\theta J(\theta) = \mathbb{E}_{s\sim\mathcal{E}} \left[\nabla_\theta \boldsymbol{\mu}_\theta(a|s) \nabla_a Q^{\boldsymbol{\mu}}(s, a)|_{a=\boldsymbol{\mu}_\theta(s)} \right] \tag{4}$$

where the action space A and the policy μ are required to be continuous for the gradient $\nabla_\theta J(\theta)$ to be valid [5].

3.2 Centralized Training

3.2.1 Multi-Agent Deep Deterministic Policy Gradient (MADDPG)

Multi-Agent Deep Deterministic Policy Gradient (MADDPG) is introduced for solving the multi-agent task where the DDPG can not work well as DDPG is designed for single-agent scenario.
For continuous state space $\boldsymbol{\mu}_{\theta_i}$, w.r.t. parameters θ_i that parametrize the policies, the gradient can be written as:

$$\nabla_{\theta_i} J(\boldsymbol{\mu}_i) = \mathbb{E}_{x,a\sim D} \left[\nabla_{\theta_i} \boldsymbol{\mu}_i(a_i|o_i) \nabla_{a_i} Q_i^\mu(x, a_1, \ldots, a_N)|_{a_i=\boldsymbol{\mu}_i(o_i)} \right] \tag{5}$$

The experience replay buffer D contains the tuples $(x, x', a_1, \ldots, a_N, r_1, \ldots, r_N)$, recording experiences of all agents. The centralized action-value function $Q_i^{\boldsymbol{\mu}}$ is updated as:

$$\mathbf{L}(\theta_i) = \mathbb{E}_{x,a,r,x'} \left[\left(Q_i^{\boldsymbol{\mu}}(x, a_1, \ldots, a_N) - y \right)^2 \right], \quad y = r_i + \gamma Q_i^{\boldsymbol{\mu}'}(x', a'_1, \ldots, a'_N)|_{a'_j=\boldsymbol{\mu}'_j(o_j)} \tag{6}$$

where $\boldsymbol{\mu}' = \{\boldsymbol{\mu}'_1, \ldots, \boldsymbol{\mu}'_{\theta'_N}\}$ is the set of target policies with delayed parameters θ'_i [6].

4 Method

4.1 Environment

For this research paper, we used the `Simple_Tag` from Multi Particle Environments (MPE) in `PettingZoo` [10]. In this version of the predator-prey game, N slower collaborative agents (predator) need to pursue M faster agents (prey) through a randomly created environment that includes L obstacles. Whenever the predator collides with the prey, they receive a reward, and the prey receives a penalty. Agents can observe the relative locations and speeds of other agents, as well as the positions of the obstacles. See Figure 6 (in appendix) for an illustration of the environment.

4.2 Experiment Setup

Our policies are parameterized by three-layer ReLU MLP with 128 units per layer. Given the limitation in computational resources and time, we trained our models for 5000 episodes and evaluated them by averaging episode rewards for the last 200 iterations. All our models managed to converge within 5000 episodes.

Environment setup details **For RQ1:** Since it only focuses on predator collaboration, we use 3 predators, 1 prey, and 2 obstacles to save computational resources. The prey has a speed that is 1.3 times faster than the predator. **For RQ2:** Since it is specifically about prey cooperation, we have adjusted the parameters accordingly: the number of prey has been increased to two, and the number of predators to four. This creates a 2:1 predator-to-prey ratio, which presents a significant challenge to the prey, who must adapt by becoming more intelligent and collaborative. The speed of both prey and predators and the number of obstacles remain the same as in the previous experiment. **For RQ3:** Experiment for this section focuses on comparison between ensemble and single training methods. To save computational resources and to eliminate differences between predators and prey using different policies, we still maintain the basic environment of 3 predators, 1 prey, and 2 obstacles, and use DDPG for both predators and prey as a baseline for comparison. The speed of both prey and predators and the number of obstacles remain the same.

4.3 Reward Shaping

This section aims to enhance the collaboration and performance of prey and predator (related to RQ1 and RQ2), we have devised reward-shaping techniques focusing on three aspects, navigation, social dynamics, and survival/hunting strategies:

`obstacle_avoidance_reward` It calculates the Euclidean distance from the agent to each obstacle and determines the minimum distance. The reward is proportional to this minimum distance, scaled by a `avoidance_factor` set to 0.1. This function affects the agent's ability to navigate safely through the environment by avoiding collisions with static obstacles.

The `collaborative_reward` function modifies agent social behavior by rewarding proximity to other agents, controlled by the `closer` flag. When `closer` is `True`, it promotes herd behavior in prey, aiding in evasion and coordination, and pack behavior in predators, enhancing trapping and hunting efficiency. When `closer` is `False`, it encourages dispersion, which for prey reduces the risk of multiple captures in a single attack, and for predators, supports strategies like encirclement by covering more area.

`incremental_distance_reward`: This function rewards prey for increasing their distance from the nearest predator. By rewarding increased distance from predators, this function directly contributes to survival tactics. On the other hand, this function encourages predators to minimise the distance to the nearest prey, promoting active hunting behaviour. This can lead to more intelligent positioning and stalking behaviours.

4.4 Training Strategies

In order to enhance collaboration between prey and predators, a comprehensive study was conducted that compared decentralised and centralised training strategies. The objective was to investigate whether centralised training could lead to an improvement in performance, in relation to research questions 1 and 2. In particular, the performance of two decentralised training strategies, DQN and DDPG, was analysed against a centralised training strategy, MADDPG. The experimental setup included six different scenarios (the first four scenarios are comparison experiment of different algo-

rithms, the fifth scenario is baseline experiment and the sixth scenario is control group): **Centralized vs. Centralized:** MADDPG (predator) vs. MADDPG (prey); **Centralized vs. Decentralized:** MADDPG (predator) vs. DDPG (prey); **Decentralized vs. Centralized:** DDPG (predator) vs. MADDPG (prey); **Decentralized vs. Decentralized:** DDPG (predator) vs. DDPG (prey); **Baseline Experiment** DQN (predator) vs. DQN (prey); **Control Group:** Random strategy (predator) vs. Random strategy (prey).

The selection of these configurations allows us to evaluate the effectiveness of MADDPG and DDPG strategies under all conditions. The baseline experiments with DQN versus DQN provide a reference for the performance of a simpler decentralised method, while the random versus random setup serves as a control to gauge the relative complexity and success of training strategies versus random actions.

4.5 Ensemble Training

Our approach utilizes an ensemble training framework inspired by policy ensembles in MADDPG [6] which is designed to enhance the robustness and adaptability of agents in a multi-agent predator-prey simulation. Each sub-policy within this framework is defined by a distinct neural network, with no shared parameters, allowing each to learn independently and develop unique strategies. All sub-policies, however, contribute to a common loss during training.

To implement ensemble training, we retained a collection of k (set to 2) sub-policies set for each agent and maintained the same policy for each agent in every episode. Actions during each episode step are determined by one of these policies, chosen randomly at the episode's start to introduce variability and enhance exploration. For evaluations, only the top-performing sub-policy is used. If we have more time, we can train with more sub-policies and use voting during the evaluation to see if this can lead to an improvement in performance.

A series of 5,000 episodes were conducted to train the prey and predator agents with the ensemble and single policies. Four different conditions were used for the experiments: **Single vs. Single**: Single policies for both predator and prey; **Single vs. Ensemble**: Single policy for predator and ensemble policy for prey; **Ensemble vs. Single**: Ensemble policy for predator and single policy for prey; **Ensemble vs. Ensemble**: Ensemble policy for both predator and prey.

5 Experiment Results

5.1 Experiments for RQ1

5.1.1 Training

Figure 1 illustrates the mean episode reward changes over episodes for various algorithms in a predator-prey system. Each curve in the figure represents a different training scenario: the label "algorithm name 1_algorithm name 2" indicates that predators are trained with algorithm 1 and prey with algorithm 2. For instance, the "MADDPG_MADDPG" curve shows both predators and prey trained using MADDPG.

Figure 1 shows that all curves converge within the first 3000 episodes, with the MADDPG vs. DDPG and MADDPG vs. MADDPG curves achieving the highest and second highest mean episode rewards, respectively. MADDPG outperforms other algorithms in training predators by efficiently capturing complex interactions in a multi-agent scenario. In contrast, DQN performs the worst for both predators and prey, yielding the lowest rewards due to its unsuitability in the non-stationary multi-agent environment. Thus, using MADDPG could enhance predator intelligence and increase rewards through strategies like group hunting.

Figure 2 displays the training results for DDPG vs DDPG using two different reward shaping techniques for predators mentioned previously. The variations in converged reward values stem from these techniques: one that rewards closer proximity between agents, resulting in positive and higher rewards, and another that penalizes proximity, leading to negative and lower rewards.

5.1.2 Evaluation

For our evaluation, we focused on measuring mean episode rewards and the number of collisions between predators and prey across 200 episodes, repeating the process 10 times to calculate an average for collisions. The mean episode rewards plots in the Appendix (see Figure 10 and 11) are somewhat noisy despite smoothed with a running average, therefore, our primary focus will be on analyzing the collision numbers, as shown in the bar plot in Figure 3.

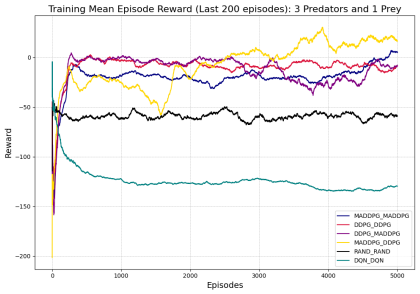


Figure 1: Training Mean Episode Reward for Different Methods (3 Predators and 1 Prey)

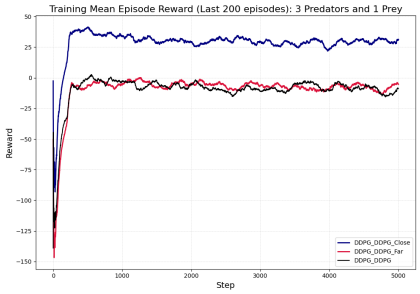


Figure 2: Training Mean Episode Reward for Different Reward Shaping Strategy (3 Predators and 1 Prey)

We can find that the MADDPG versus DDPG gives the highest collision numbers while the DQN versus DQN gives the lowest collision numbers in our evaluation which indicates that MADDPG gives the predator the strongest learning ability to learn from the other agents and the past so the predator with MADDPG algorithm has the highest collision numbers. This is because the MADDPG algorithm is able to capture complex interactions between agents in this multi-agent predator-prey system. However, the DQN equips the agent with the weakest learning capacity and ends with the smallest collision numbers due to the non-stationary multi-agent environment.
In the blue bars, we compare the number of collisions per episode between DDPG-trained predators with two different reward-shaping strategies and DDPG-trained prey. We observe that the reward-shaping strategy that encourages proximity results in better performance compared to the strategy with no reward shaping, which is shown in the third blue bar. The reward shaping strategy that discourages proximity is the least effective, performing worse than both the strategy that encourages proximity and the one with no reward shaping. This indicates that the strategy encouraging proximity is more effective for the predator agents and is more likely to simulate intelligent predator behaviors such as group hunting. On the other hand, the strategy that discourages proximity does not lead to an increase in collisions per episode, suggesting that in scenarios with only one prey, predators collaborating closely is more beneficial than spreading out.

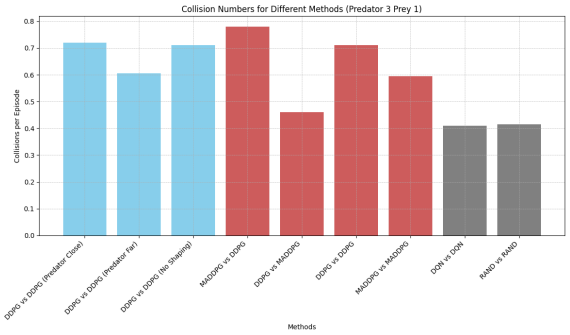


Figure 3: Collision Numbers for Different Training Methods (Predator 3 vs Prey 1)

5.2 Experiments for RQ2

5.2.1 Training

As illustrated in Figure 7 (in Appendix), all experiments successfully converged within 5000 episodes. Notably, the experiments where agents were trained using MADDPG versus MADDPG, DDPG versus DDPG, and MADDPG versus DDPG demonstrated higher mean rewards, particularly for predators trained with MADDPG (represented by the yellow and blue lines), which achieved the highest rewards. This outcome is expected, given that these predators were trained using a more advanced method (MADDPG), leading to higher overall rewards. It's important to note that the rewards were calculated collectively for both prey and predators.

In contrast, experiments with DDPG versus MADDPG and RAND versus RAND converged to lower mean rewards, indicating that DDPG-trained predators are less effective at capturing MADDPG-trained prey compared to other training configurations. The DQN versus DQN setup exhibited the lowest mean rewards because the DQN algorithm may struggle with the non-stationary environments presented in these multi-agent interactions.

Figure 8 (in Appendix) displays the training results for DDPG vs DDPG using two different reward shaping techniques for prey. The variations are due to similar reasons from the previous section.

Regarding the speed of convergence, all training methods displayed similar patterns, with convergence typically occurring around 1000 episodes.

5.2.2 Evaluation

Figure 9 (in Appendix) is the bar chart that shows the number of collisions per episode for different algorithms. The red bar represents the collision numbers in the comparison experiments, the blue bar represents the collision number of algorithms with reward shaping and the grey bar represents the collision numbers in the baseline experiment.

The blue bars in the graph represent the results of using reward-shaping strategies (either encouraging or discouraging prey to stay close together). The results show that when prey proximity is discouraged, the number of collisions per episode decreases. When prey are encouraged to stay closer together, the number of collisions is further reduced.

The red bars on the graph represent the results of using various training strategies for predator vs prey. Notably, training prey using the MADDPG approach results in fewer collisions. This supports our earlier discussion about how a centralized critic mechanism in MADDPG may foster better collaboration and can enhance performance among agents. Conversely, when prey are trained using the DDPG, collision numbers are higher, especially when the predator is trained with MADDPG.

The grey bars show the baseline results, with the DQN showing the lowest number of collisions. This may be due to both prey and predator may not have developed intelligent behaviors when trained with DQN, therefore showing a lower collision than random.

5.3 Experiments for RQ3

5.3.1 Training

Figure 4 illustrates the trajectories of the average rewards per episode for agents trained under five distinct ensemble setups, as discussed earlier. These average rewards are calculated over the last 200 episodes. Each experiment runs for 5000 episodes, and all four training setups successfully converge. The data show that four of the conditions resulted in better reward trajectories compared to taking random actions. Notably, the 'SINGLE vs SINGLE' training setup converged at a lower average reward compared to the other curves. This outcome is reasonable, as 'SINGLE vs SINGLE' likely involves less complex strategies during training, which can limit the learning capabilities of the agents. However, this still needs further analysis in the evaluation stage to make a better conclusion.

5.3.2 Evaluation

For evaluation, we plotted mean episode rewards (see Fig 12 in Appendix) and the collision number between predators and prey over 200 episodes (see Fig 5) which averaged over 10 runs. For the same reason discussed previously, our analysis will focus on the collision number.

In Figure 5, the scenario where predators used ensemble training while prey used a single policy resulted in the highest number of collisions ('ENS vs SINGLE'). When both predators and prey were trained using ensemble methods, the collision number was the second highest ('SINGLE vs ENS'). This observation is consistent with the findings from Lowe et al. [6], and suggests that ensemble training is an effective strategy.

When predators were trained with a single policy, the number of collisions was similar whether prey was trained with a single policy or with an ensemble of policies. Given the high predator-prey ratio of 3:1, this result is reasonable as prey remains in a challenging position even when trained with multiple policies. With more time, further investigation of different predator-prey ratios may

provide additional insight. It's worth noting that all four training setups resulted in higher collision frequencies compared to the baseline.

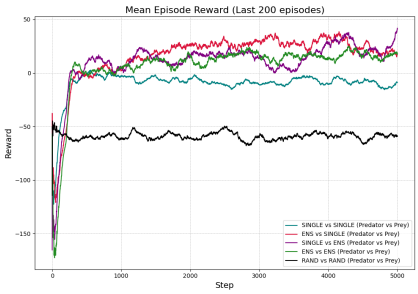


Figure 4: Ensemble Training Mean Episode Reward Comparison

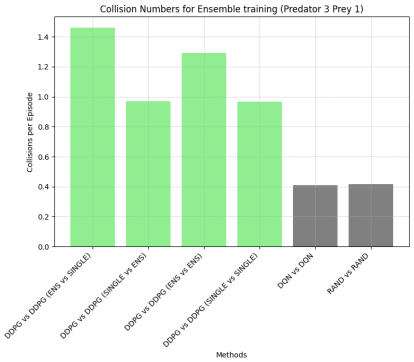


Figure 5: Collision number for ensemble training

6 Discussion and Conclusion

6.1 What do the Results say about the Research Questions

The first two research questions investigate whether collaborative behavior among predators and prey enhances their performance. Our findings suggest that reward structures promoting teamwork—such as keeping predators or prey closer together—result in more successful interactions (i.e., more collisions for predators and fewer for prey). Furthermore, we found that using a centralized training approach (MADDPG) leads to significantly better outcomes than a decentralized method (DDPG) for both predators and prey. These results confirm that fostering cooperation leads to enhanced performance, with both reward shaping and centralized training improving outcomes, particularly the latter. The results are aligned with results in the paper [4], [6].

The third research question focuses on how to avoid overfitting to behaviour of competitors leading to worse generalized performance. Our experiments demonstrate that agents training with an ensemble of sub-policies significantly outperform the agents training on a single policy strategy because the ensemble method mitigates the risk of overfitting and offers a richer policy selection for agents in every episode. The result is also aligned with the paper [6].

6.2 Limitation and Opportunity for Future Work

Exploration of Predator-Prey Ratios Our research was confined to examining only two specific predator-prey ratios due to limitations in computational resources and time. Future studies could explore scenarios with significantly more prey than predators. This could lead to different outcomes, such as whether spreading predators to encircle prey is more effective than clustering them together, a strategy we could not fully evaluate, or swarming behaviour.

Duration of Training Episodes Our training episodes are limited to 5000 episodes. Extending these could offer more comprehensive learning opportunities, allowing models to develop complex strategies and adapt to varied scenarios. This extension could enhance their applicability in real-world settings. The original paper on MADDPG utilized up to 25,000 episodes, suggesting room for extensive training ([6]).

Ensemble Training Approaches Our methodology involved using an ensemble of two sub-policies, and selecting the more effective one during evaluations. This approach may limit overall performance. With additional resources, implementing a larger ensemble with more sub-policies and employing a majority voting mechanism could potentially yield better outcomes.

Neural Network Architecture of Policy The current policy uses MLP architecture that occasionally suffers from gradient vanishing, which we mitigate using gradient clipping. Given more time, experimenting with activation functions such as LeakyRELU instead of RELU, or exploring alternative architectures, could potentially mitigate these issues and improve model training.

References

[1] Manuele Brambilla, Eliseo Ferrante, M. Birattari, and Marco Dorigo. Swarm robotics: a review from the swarm engineering perspective. *Swarm Intelligence*, 7:1 – 41, 2013.

[2] SAM DEVLIN, DANIEL KUDENKO, and MAREK GRZEŚ. An empirical study of potential-based reward shaping and advice in complex, multi-agent systems. *Advances in Complex Systems*, 14(02):251–278, 2011.

[3] Érika Diz-Pita and M. Victoria Otero-Espinar. Predator–prey models: A review of some recent advances. *Mathematics*, 9(15), 2021.

[4] Marek Grzes. Reward shaping in episodic reinforcement learning. 2017.

[5] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[6] Ryan Lowe, Yi I Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in neural information processing systems*, 30, 2017.

[7] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning, 2013.

[8] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[9] Ming Tan. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proceedings of the tenth international conference on machine learning*, pages 330–337, 1993.

[10] Jordan Terry, Benjamin Black, Nathaniel Grammel, Mario Jayakumar, Ananth Hari, Ryan Sullivan, Luis S Santos, Clemens Dieffendahl, Caroline Horsch, Rodrigo Perez-Vicente, et al. Pettingzoo: Gym for multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, 34:15032–15043, 2021.

A Appendix

A.1 Additional Figures

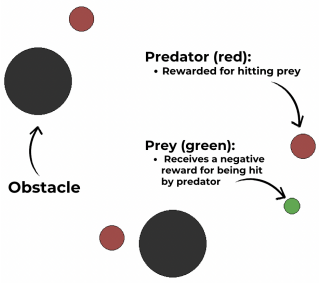


Figure 6: Illustration of `Simple_Tag` Environment

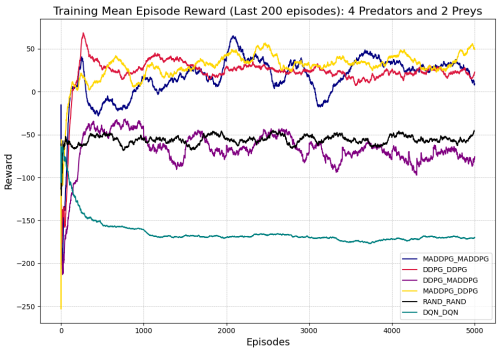


Figure 7: Training Results for Different Methods (4 Predators vs 2 Prey)

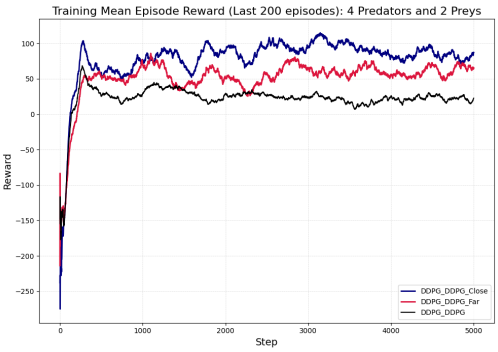


Figure 8: Training Results for Different Reward Shaping (4 Predators vs 2 Prey)

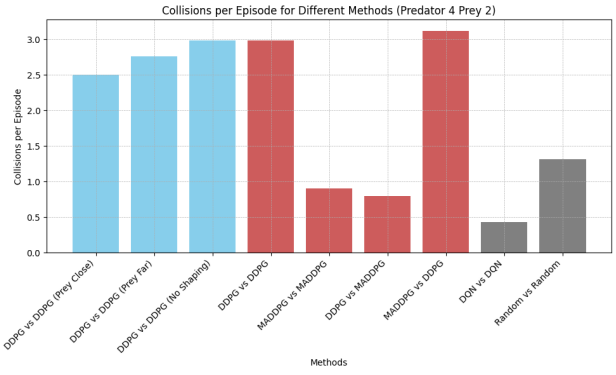


Figure 9: Collisions per Episode (Predator 4 Prey 2)

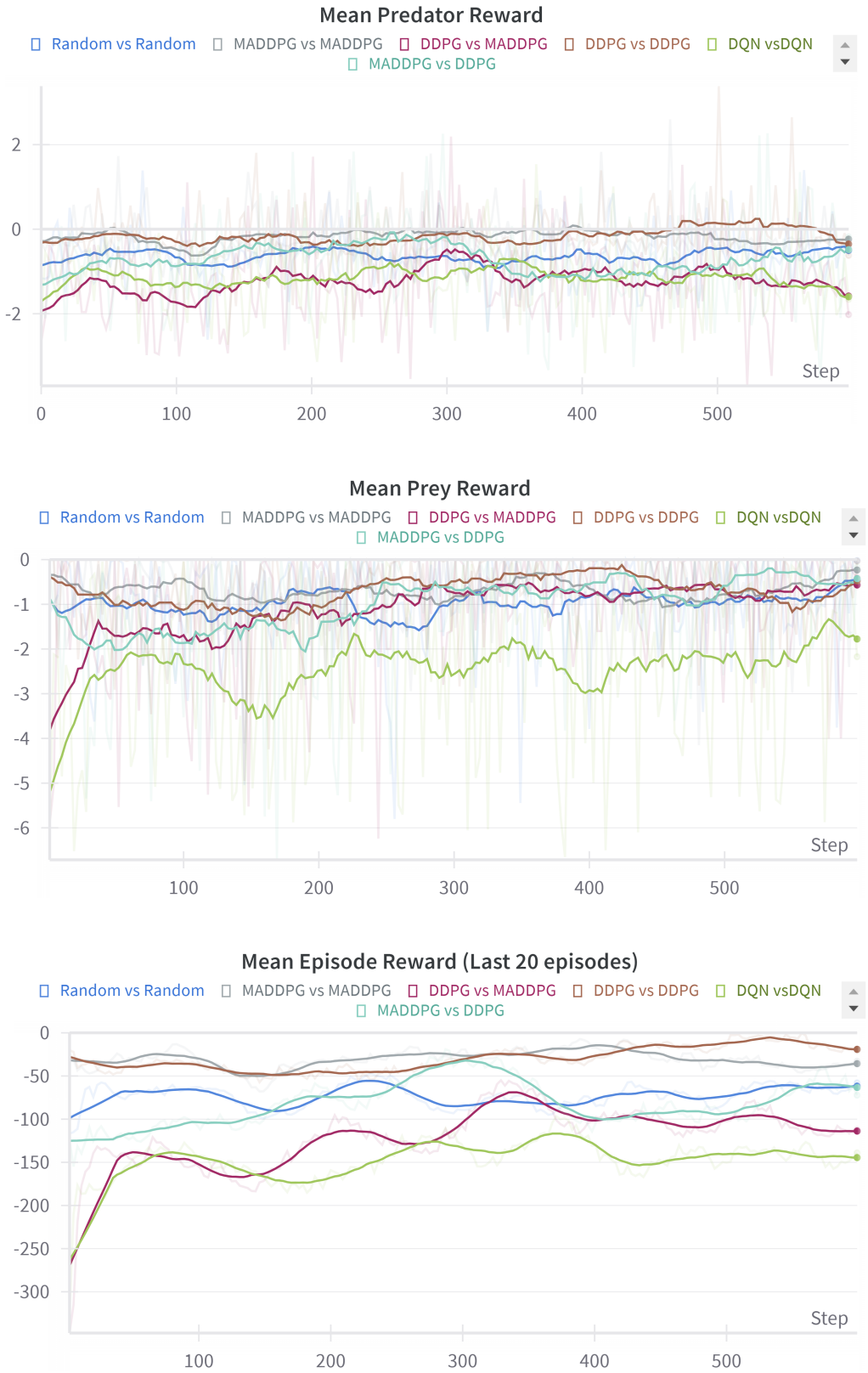
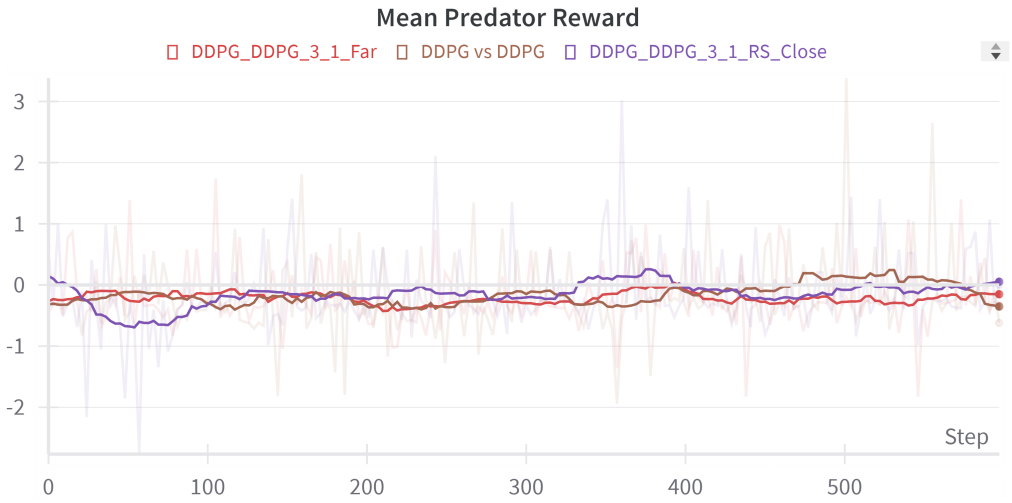
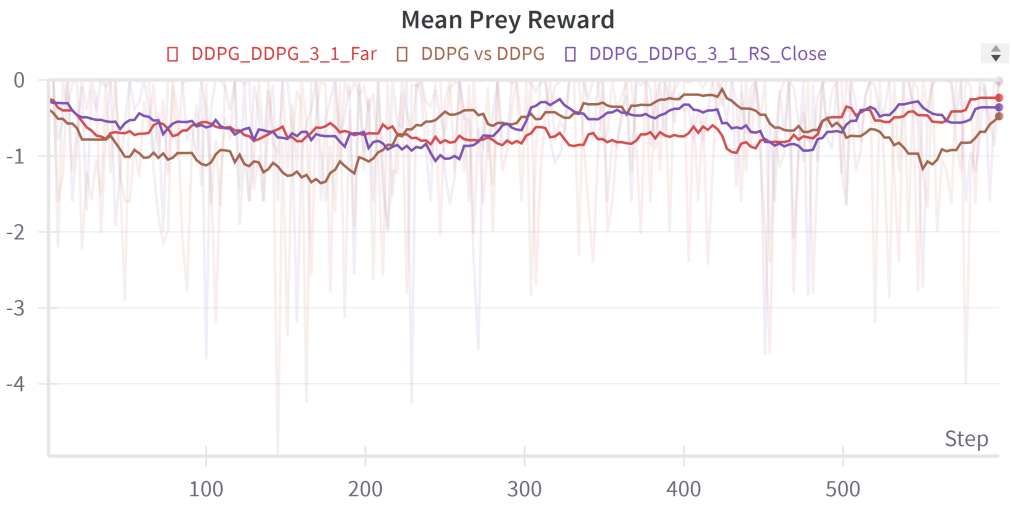


Figure 10: Overview of Evaluation of Predator 3 and Prey 1 (Different Training Methods)





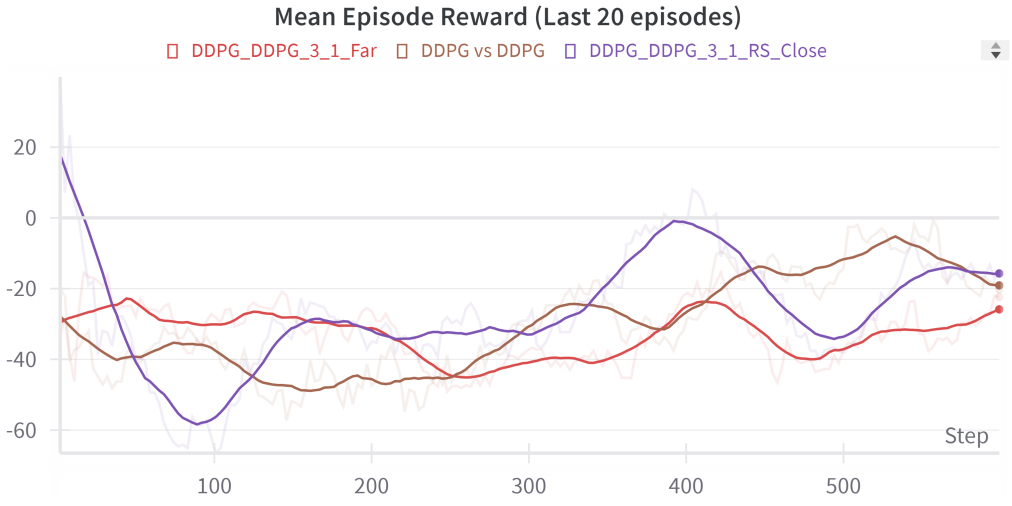
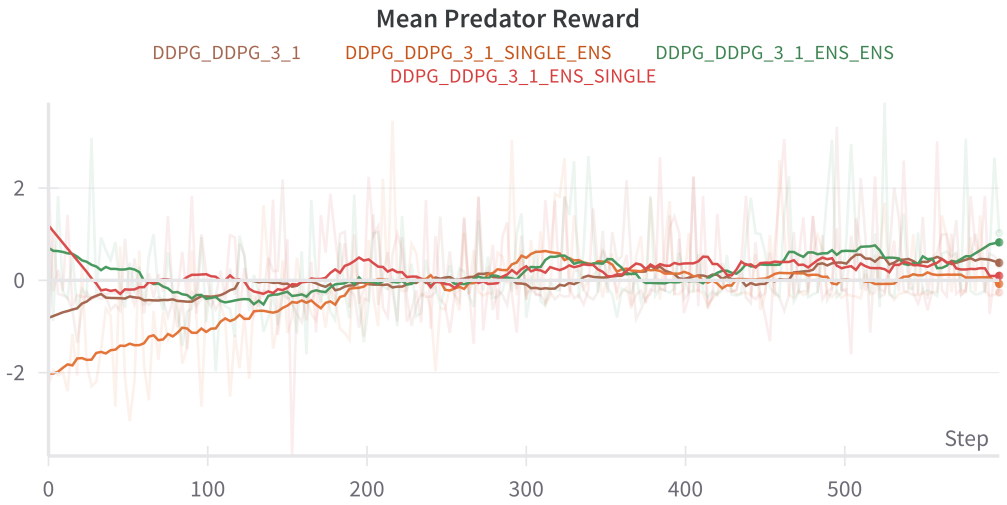
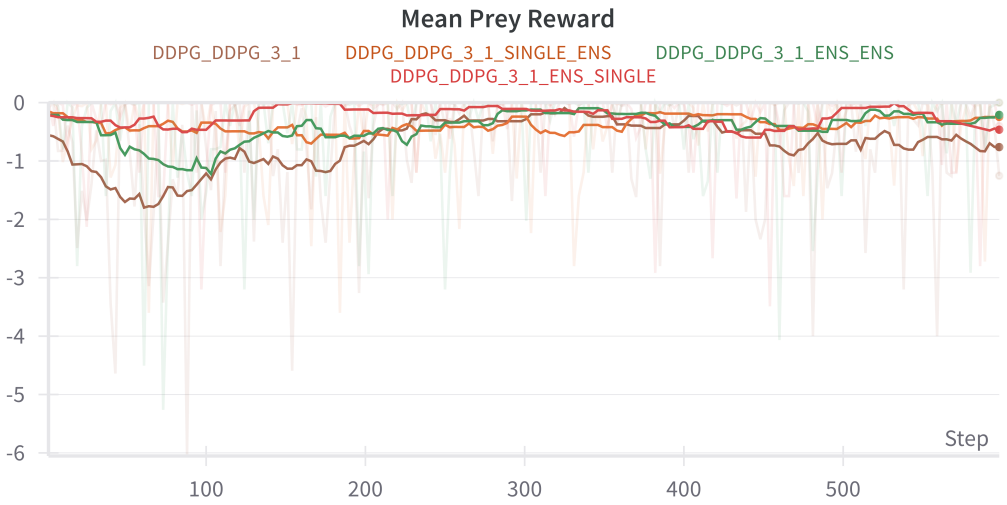


Figure 11: Overview of Evaluation of Predator 3 and Prey 1 (Reward Shaping)





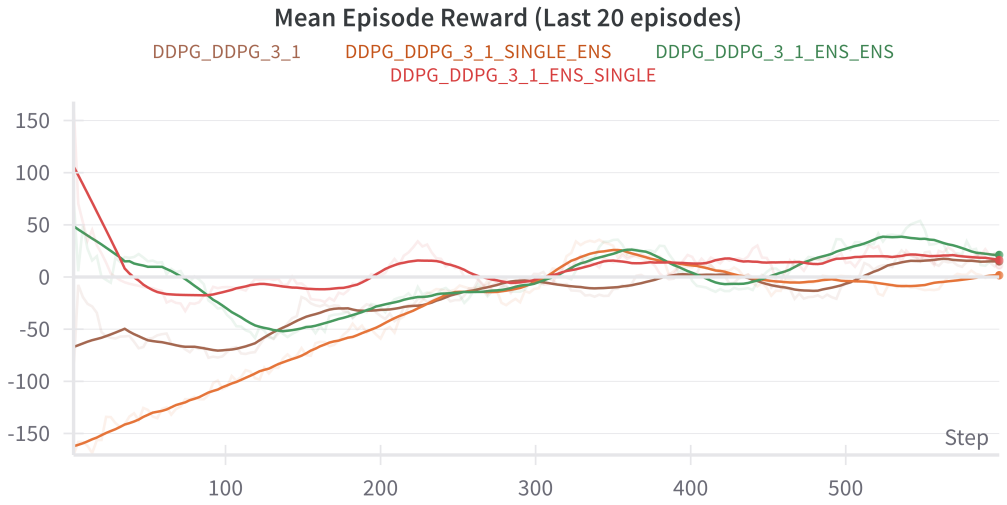


Figure 12: Overview of Evaluation of Predator 3 and Prey 1 (Ensemble Method)

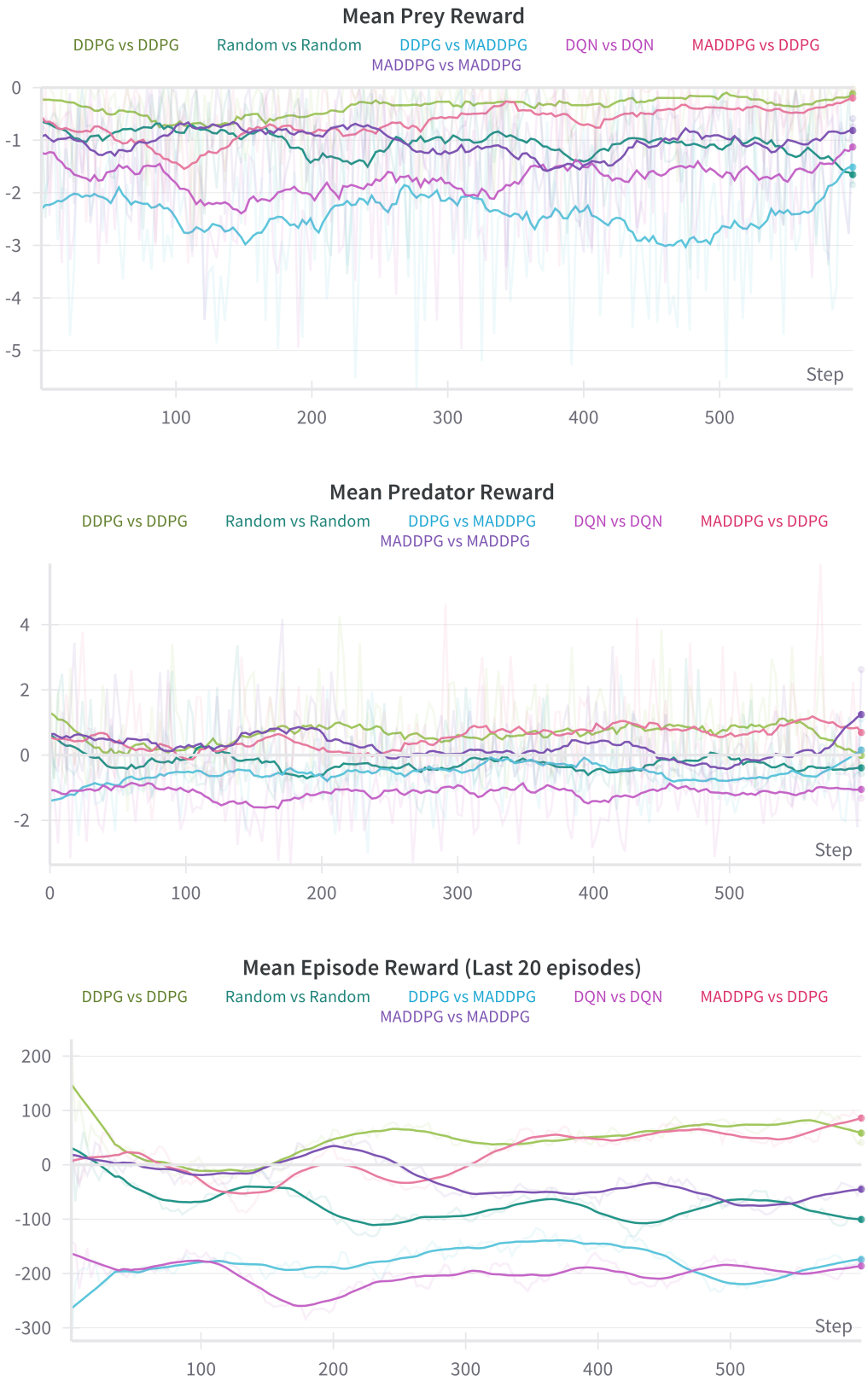


Figure 13: Overview of Evaluation of Predator 4 and Prey 2 (Different Training Methods)

Figure 14: Overview of Evaluation of Predator 4 and Prey 2 (Reward Shaping)