Benchmarking Goal-Conditioned Reinforcement Learning with Soccer

Authors. Venkat Subramanian (venkat.subra@princeton.edu), Anthony Zhai (zhai@princeton.edu), Sai Jogannagari (sai.jogannagari@princeton.edu)

Link to Github Repo. https://github.com/antz22/COS435-GCRL_MARL/tree/submission (submission branch contains the core of the code we used, other branches contain rest of the project code we didn't directly use but are necessary to run the code).

1 Introduction

Reinforcement learning (RL) has previously had great success in discovering novel strategies to solve complex tasks, most notably in games such as Go, Atari, and Chess [5–7]. However, complex agents staged in physics-based environments pose their own set of challenges, as training a good policy requires coordinated movement in a large state space [3]. Framing the task with a sparse reward signal such that only a successful completion of the task receives positive reward (goal-conditioned reinforcement learning) further complicates this task, although it offers the benefit of not restricting how the agent should learn [1].

Existing goal-conditioned physics-based environments that entail the training of robotic agents are limited to tasks such as moving to a target location, pushing a large external object (which does not require much precision), or pushing a smaller object for relatively short distances [2]. This paper aims to establish a new, more complex environment for training an 'ant' quadruped agent to push a ball over a large distance into a target goal, with an alternate configuration of the environment containing an obstacle in front of the goal. By creating more difficult environments and determining which, if any, existing RL algorithms successfully learn robust policies, we can spark new research on RL algorithms that ideally also work to solve other similarly difficult problems.

We built our environment as an extension of the implementations of the Ant and AntBall environments from the JaxGCRL repository [2]. In the Ant environment, a quadruped 'ant' agent with 8 joints (4 hip joints and 4 ankle joints) must learn how to move to a randomized target location. In the AntBall environment, this same quadruped 'ant' agent must learn how to move a ball into a target hemisphere 5 units away. In both of these environments, the optimal policy for the ant involves coordinating its joints to either move to a target location or to move an external object to a target location. Our benchmark extends the AntBall environment in 2 aspects to necessitate the development of more complex strategies: increased distance and incorporation of an obstacle.

We introduce two versions of a soccer-based ant ball environment: AntSoccerV1 and AntSoccerV2. The first version (AntSoccerV1) simply visually remodels the target hemisphere into a goal with a rectangular target area and establishes the target goal to be 10 units away from the ant (doubling the distance from the AntBall environment). The second version (AntSoccerV2) introduces boundaries around the field, a rectangular obstacle in front of the goal to simulate a static defender blocking the goal, a larger goal, and an increased distance of the goal to 20 units away from the ant.

Our new benchmarks provide a unique challenge because the increased distance between the ball and the target adds complexity to the ball-pushing task. Additionally, adding an obstacle further restricts the domain of possible strategies for achieving the goal (and eliminates the most direct strategy of the agent pushing the ball directly forward). AntSoccerV1 acts as an intermediary environment to measure the performance of an algorithm in learning to kick a ball into a goal from farther away, whereas AntSoccerV2 is a significantly more challenging environment which measures the performance of an RL agent in performing a multitude of complex skills used in soccer.

Our work demonstrates that the AntSoccerV2 environment is significantly harder than the AntBall environment, as existing single-agent RL algorithms like PPO and CRL and multi-agent RL algorithms like ICRL are not able to learn effectively on this new environment. This prompts the development of new RL algorithms to test their success on this new, more difficult environment.

This report is organized as follows: first, we describe prior work that has been done to tackle the problem of the Ant and Ant Ball environments. Next, we detail the experiments we ran on the Ant and Ant Ball environments to reproduce and verify previous results. We then describe the experiments we ran on the AntSoccerV1 and AntSoccerV2 environments to test PPO, CRL, and (multi-agent) ICRL performance. Next, we describe the results based on training graphs and reward based on goals scored. Finally, we discuss the implications of our work and potential limitations.

2 Prior Work

Our work mainly builds upon the goal-conditioned reinforcement learning paradigm with the use of contrastive learning to solve tasks in complex environments [1]. Various algorithms have been employed to train agents to solve similar tasks in physics-based simulation environments, including proximal policy optimization (PPO), contrastive reinforcement learning (CRL), and independent contrastive reinforcement learning (ICRL) [1, 2]. CRL and ICRL in particular offer an interesting approach to tackling our environment, as it allows the agent (in single and multi-agent formulations) to learn an encoded representation of the complicated state-action space and goal states. This potentially improves ease of learning even with complicated dynamics since finding an optimal policy in the regular state space may be hard, but the optimal policy in the encoded space just involves taking actions that minimize distance to the encoded goal state.

Prior research on training RL agents to play soccer focus more on matching human performance, using behavioral cloning with human expert data to train RL agents to play 2v2 simulation-based matches [4]. However, these learned policies are limited in only being able to replicate preexisting human strategies for soccer as opposed to being able to learn completely novel strategies with no human input, which is the aim of our paper.

3 Methods

3.1 Environments

Our work is based off of the Ant environment developed by Google [3] and the AntBall environment developed by Bortkiewicz et. al [2]. Both of these environments utilize the Brax physics engine [3] and an ant agent with 4 legs consisting of 4 hip joints and 4 ankle joints. In the Ant environment, the target location is an upper hemisphere of radius 0.9 that the ant must move into to achieve the defined goal. In the AntBall environment, the ant must instead move a ball with a radius of 0.5 units into the same upper-hemisphere target. Training these environments with multi-agent algorithms required developing a multi-agent compatible wrapper of AntSoccerV1 and AntSoccerV2, based on an existing multi-agent wrapper of the Ant environment.

AntSoccerV1 extends AntBall by reshaping the target location from a hemisphere to a 4x1x2 rectangular box resembling that of a goal. The distance of the target is doubled from 5 units to 10 units, and the location of the goal is fixed (whereas in AntBall the target location is at a random angle with respect to the ant). Additional cylindrical rigid posts are present at the edges of the target box to resemble goalposts. AntSoccerV2 further extends AntSoccerV1 by again doubling the distance to 20, enlarging the goal, adding rigid boundaries to the soccer field, and adding a rectangular obstacle in front of the goal. All 4 environments are shown in 1.

We additionally experimented with modifications to the AntSoccerV2 environment to make it slightly simpler. First, we fixed the ball position so it is not randomly placed, as we noticed that in many iterations, the agent would run up to the ball in a spot that it "expected" the ball to be in but the ball would not be there due to randomized placement. We also tried turning off the rigidity of the boundaries next to the goals to make the environment slightly easier. These two aspects are modifiable and can be reverted to make the environment harder. the environment also allows easily increasing the width of the obstacle to represent a "larger" goalie.

3.2 Experiments

We began with training PPO, CRL, and ICRL on the Ant environment to reproduce the results from [1] and to compare the performance of contrastive RL algorithms (CRL and ICRL) with RL algorithms that typically rely on a reward function, such as PPO. In order to compare training time and number of training steps, we trained these algorithms on the same number of learning steps to verify if the ICRL algorithm with the multi-agent formulation of Ant trained in less steps than the CRL algorithm with the default single-agent formulation [1]. In all environments, we trained with an episode length of 1001 and otherwise default parameters as configured in repositories we extended. Number of training iterations varied between environments, but we generally trained until success stabilized, a noticeable difference between algorithms was noticed, or long enough had passed with no success obtained for a given algorithm (and with loss functions converging). Training iterations for ICRL in later environments was also limited due to slow real world training time (code obtained from a different repository than CRL and PPO).

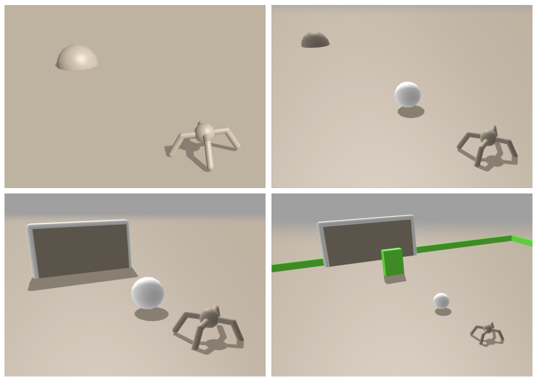


Figure 1: Ant and AntBall environments shown from left to right in the top row. AntSoccerV1 and AntSoccerV2 environments shown from left to right in the bottom row. With increasing environmental complexity, there is a larger range of possible strategies for the agent to learn from which simultaneously makes training more difficult.

After obtaining preliminary results on the Ant environment we developed the AntSoccerV1 environment, which closely matched the AntBall environment with slight visual modifications. In order to apply ICRL to AntSoccerV1 we developed the code for training an ant agent on the AntSoccerV1 environment where each joint served as its own independently-trained agent, based off of work from the previous implementation of the code to train an ant with ICRL on the Ant environment [1]. We additionally trained CRL on AntSoccerV1 and trained two versions of PPO: one without modifications from the JaxGCRL repository [2] and one with modifications to the reward function. Both reward functions are specified in the section 3.3.

Preliminary results from these training experiments led us to extend the environment to AntSoccerV2, which added more distance between the goal and the ant, an obstacle in front of the goal, and a boundary to the soccer field. We trained the same algorithms (PPO, CRL, and ICRL) on AntSoccerV2 to directly compare performance of these methods with the AntSoccerV1 environment.

3.3 Reward Functions

We use the following definitions for each of the reward functions formulated in this section:

B_t = (x,y) position of Ball at time t,
g = (x,y) position of goal,
A_t = (x,y) position of Ant at time t,
A_z^t = z-position of Ant at time t,
a_t = action at time t.

In the provided Ant environment, the dense reward function used by PPO was the following:

$$r_t^{\text{dense}} = 10\frac{|A_{t-1} - g| - |A_t - g|}{dt} + \mathbb{1}_{\{0.3<A_z^t<1\}} - 0.5||a_t||^2 \tag{1}$$

Intuitively the first term of the reward function denotes rewarding the ant for moving closer to the goal, the second term denotes rewarding the ant for staying upright, and the last term encourages the ant to take actions that have lower magnitude.

The sparse reward function utilized by CRL, ICRL, and PPO is given by the following:

$$r_t^{\text{sparse}} = \mathbb{1}_{\{||B_t-g||<0.5\}} \tag{2}$$

This gives a reward to the ant if the ball is within 0.5 units of the goal.

The provided dense reward function in AntBall is given by:

$$r_t = 10 \cdot \frac{||B_{t-1} - g|| - ||B_t - g||}{\text{dt}} + \mathbb{1}_{\{0.2<A_z^t<1\}} - 0.5||a_t||^2 \tag{3}$$

Intuitively, the first term $10 \cdot \frac{||B_{t-1}-g||-||B_t-g||}{\text{dt}}$ rewards the ball getting closer to the goal, the second term $\mathbb{1}_{\{0.2<A_z^t<1\}}$ rewards the ant for staying upright, and the last term $-0.5||a_t||^2$ imposes a penalty based on the magnitude of the ant's action at time t.

We also introduced our own modified dense reward function for AntSoccerV1, which is given below:

$$R_{t+1} = 10 \cdot \mathbb{1}_{\{||B_t-g||<0.5\}} + \mathbb{1}_{\{0.3<A_z^t<1\}} + 2 \cdot (||A_{t-1} - B_{t-1}|| - ||A_t - B_t||) \tag{4}$$
$$+ \frac{5}{||B_t - G||} \cdot (||B_{t-1} - G|| - ||B_t - G||) \tag{5}$$

Intuitively, the first term $10 \cdot \mathbb{1}_{\{||B_t-g||<0.5\}}$ involves providing a reward for when the ball is close to the goal, the second term $\mathbb{1}_{\{0.3<A_z^t<1\}}$ provides a reward for the ant staying upright, but raises the lower bound for getting this reward to be 0.3 instead of 0.2 in 3. The third term $2 \cdot (||A_{t-1} - B_{t-1}|| - ||A_t - B_t||)$ rewards the ant for moving closer to the ball, and the fourth term $\frac{5}{||B_t-G||} \cdot (||B_{t-1} - G|| - ||B_t - G||)$ rewards the ant for moving the ball closer to the goal.

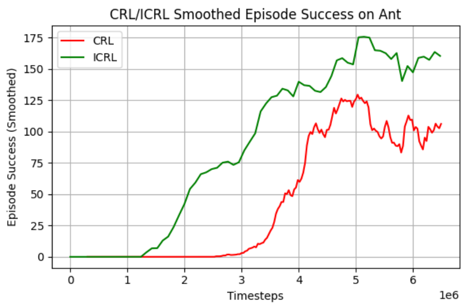
For AntSoccerV2, we slightly modified the dense reward function for PPO from 1, boosting the coefficient on $\frac{1}{||B_t-G||} \cdot (||B_{t-1} - G|| - ||B_t - G||)$ to be 10 instead of 5. The intuition behind this was that, since the distance between the ant, ball, and goal are increased in AntSoccerV2 from AntSoccerV1, it was necessary to boost the reward for the ball moving closer to the goal:

$$R_{t+1} = 10 \cdot \mathbb{1}_{\{||B_t-g||<0.5\}} + \mathbb{1}_{\{0.3<A_z^t<1\}} + 2 \cdot (||A_{t-1} - B_{t-1}|| - ||A_t - B_t||) \tag{6}$$
$$+ \frac{10}{||B_t - G||} \cdot (||B_{t-1} - G|| - ||B_t - G||) \tag{7}$$

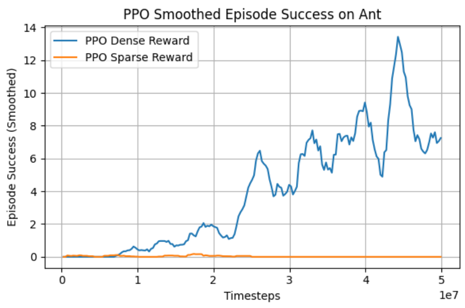
4 Results

4.1 Ant

The results of training CRL and ICRL on the Ant environment are shown in 2a, and results of training dense and sparse reward PPO are show in 2b. Just as the authors of [1]



(a) Performance of CRL and ICRL on the existing Ant environment



(b) Performance of PPO with a dense and sparse reward on the existing Ant environment

Figure 2: Comparing single-agent (CRL) and multi-agent (ICRL) contrastive learning methods with PPO. All graphs were generated by smoothing episode success over 8 time bins. The contrastive learning methods were trained for 6,500,500 iterations, whereas the PPO methods were trained for 50,000,000 iterations. Notice the difference in scales in the episode scales between the contrastive learning and PPO methods on the Ant environment.

find, we see in 2a that ICRL converges "faster" on the Ant environment than CRL. Recall that the difference between ICRL and CRL is that ICRL independently trains a policy for each individual leg, while CRL trains a policy for all 4 legs at once. This indicates that multi-agent algorithms are not necessarily harder to train than single-agent algorithms for the same task, confirming a prior result [1]. Upon training, we noticed that CRL and ICRL, both of which used a sparse reward signal of only getting rewarded when reaching the goal, reached the target location in approximately 100-150 episodes out of 1001 by less than 6.5 million training iterations. however, ICRL converges quicker and performs better than CRL.

In 2b we see that PPO with the dense reward function from 1 took a significantly longer time (approximately 50 million iterations) to get approximately 0.1 times the reward that CRL and ICRL received. Furthermore, when PPO was trained on the sparse reward signal seen in 2, it received near 0 reward.

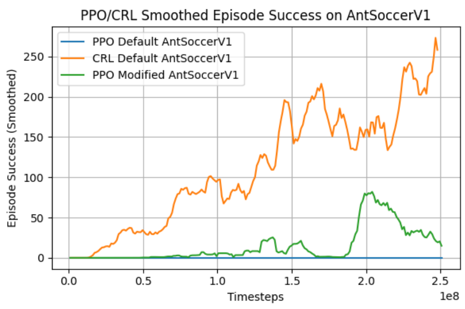
4.2 AntSoccerV1

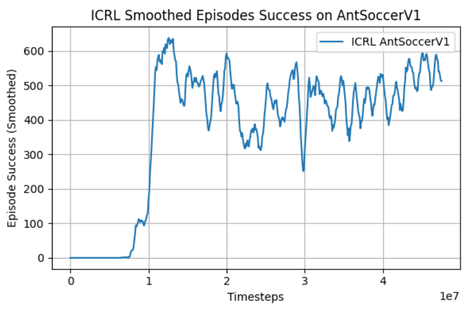
Next, in the AntSoccerV1 environment, we trained the same CRL and ICRL algorithms to analyze whether ICRL was able to learn faster and more effectively than CRL in this environment as well. We also trained PPO using the out-of-the-box reward function provided in the JaxGCRL repository for AntBall [2], and we also trained PPO using our modified dense reward function(4). The training curves are shown in 3.

In 3a, we see that PPO using the dense reward function provided in the existing repository (3) does not learn at all on the AntSoccerV1 environment. Even with a highly engineered reward function for the AntSoccerV1 environment that rewards the ant for getting close to the ball and rewarding the ant for bringing the ball closer to the goal(4), the performance of PPO is worse than CRL, which hits successes for up to 250 episodes out of 1001.

Furthermore, we noticed that renderings of the PPO and CRL algorithms were very different on successful iterations. Neither algorithm was penalized for "control costs," or making too many actions. However, PPO used a "dribbling" strategy, where the ant agent would essentially "hug" the ball and vibrate its back legs to push it slowly towards the goal (renderings are provided in the wandb.ai links in charts.py in our data analysis code). On the other hand, CRL learned a policy that resembled more of a "kicking" strategy in which the agent more forcefully kicked the ball into the goal in longer, coordinated movements.

We also see that ICRL significantly outperforms the other methods on this environment, reaching successes of upwards of 400 in just a few million iterations of training. This mirrors the results from the Ant environment, where ICRL also trained faster and reached higher success than CRL. However, we are slightly skeptical of these results because we





(a) Performance of PPO and CRL on the AntSoccerV1 environment.

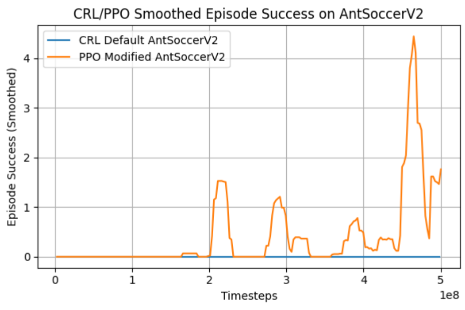
(b) Performance of ICRL on the AntSoccerV1 environment.

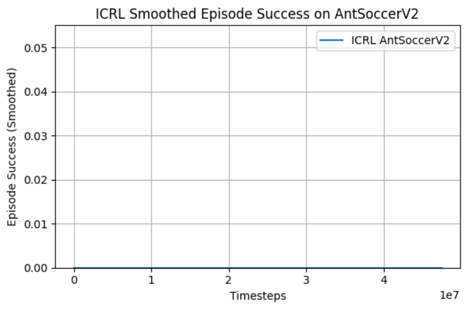
Figure 3: Comparing single-agent and multi-agent RL methods on the AntSoccerV1 environment. All graphs were generated by smoothing episode success over 8 time bins. PPO and CRL were trained for 250,000,000 iterations, whereas ICRL was trained for 5,000,000 iterations. Notice the difference in scales between the two charts.

were not able to render visualizations and may have had an implementation bug that caused unusually fast and successful training (which will be discussed more in a later section).

4.3 AntSoccerV2

Results for training CRL, ICRL, and PPO with dense reward on the AntSoccerV2 environment (with the aforementioned modifications in section 3.1 to make it slightly easier) are shown in 4. Here, we can see that our highly engineered dense reward function from 6 is partially able to generalize to the harder environment, but the ant is still not able to effectively get the ball into the goal consistently even after 500M timesteps. In the iterations where PPO did learn to get the ball into the goal, it exploited the lack of boundaries to the left and right of the goal and drove the ball behind the goal, veering away from the obstacle in front of the goal. We also noticed that the strategy that PPO took on successful iterations was very similar to the strategy it took in AntSoccerV1, which involved the ant staying very close to the ball and "dribbling" it to the goal. Furthermore, since we took out the control costs in our dense reward functions 4 and 6 from 3, we noticed that the ant also had high control costs and was visibly "vibrating" as moved the ball.





(a) Performance of PPO and CRL on the AntSoccerV2 environment.

(b) Performance of ICRL on the AntSoccerV2 environment.

Figure 4: Comparing single-agent and multi-agent RL methods on the AntSoccerV2 environment. Episode successes were smoothed over 8 time bins. PPO and CRL were trained for 500 million iterations. ICRL was only trained for 50 million iterations, due to no successes obtained even after some loss function stabilization and extremely slow real world training time.

We hypothesize that CRL and ICRL did not learn on this environment since the distance was too far for the algorithms to encode a representation space, and the obstacle was also blocking many easy paths to the goal. Contrastive learning and goal-conditioned methods depend on learning strategies that are "aligned" to leading to the goal state, but if the goal state or states close to the goal state are extremely unlikely to be reached, then it is very difficult to make any improvements to the policy. We think that this failure to reach states close to the goal state causes the behavior we observe of the ant simply staying still in the starting position.

ICRL also struggles to learn on AntSoccerV2. We see that after 50 million time steps, some loss functions converge but still no goals have been scored. Although we were not able to reach 500 million time steps as we did for PPO and CRL (due to the longer training time per iteration for ICRL), we hypothesize that the algorithm would not significantly affect performance because the state and action space still remain incredibly large and CRL is also not able to learn by 500 million time steps.

Given the limited success of PPO and contrastive learning methods, we see that AntSoccerV2 is a very difficult environment which requires further innovations in constructing complex reward functions or developing new algorithms that are able to adapt to large state spaces in physics-based environments.

5 Discussion and Limitations

Although our paper has worked to introduce a new benchmark for goal-conditioned RL, the algorithms that we have tested on the AntSoccerV2 environment have been mostly unsuccessful, while the algorithms tested on AntSoccerV1 have had mixed success. Future work will be focused on developing robust algorithms for learning the complex and coordination-based strategies required for succeeding in the AntSoccerV2 environment. Further work to confirm the results of ICRL on AntSoccerV1 is also needed, as the difference between ICRL and CRL on AntSoccerV1 is much larger than the difference between those contrastive learning algorithms on Ant. A potential implementation issue with extra successes mistakenly being awarded in the multi-agent environment or insufficient position randomization could also be causing ICRL to train faster and better than it really should. To verify this we can render agents trained with ICRL. This is not yet done due to bugs faced when saving policy parameters and rendering parameters in code from the anonymous repository we used for training with ICRL. Fixing these issues and successfully rendering ICRL would also allow us to analyze the performance of ICRL qualitatively.

Another limitation of our work is that we have only tested a limited suite of algorithms on our environments, excluding many common RL algorithms such as Soft Actor-Critic (SAC) and Twin Delayed Deep Deterministic Policy Gradient (TD3). We also only tested one multi-agent algorithm (ICRL) on our environments. We determined that the multi-agent versions of PPO, such as Independent Proximal Policy Optimization (IPPO) and Multi-Agent Proximal Policy Optimization (MAPPO), would also likely be poor choices for the environments in the goal-conditioned formulation, due to the sparse reward. However, we did not test any algorithms suited for goal-conditioned environments that involve information sharing between multiple agents during training and independent action execution, normally called Centralized Training with Decentralized Execution (CTDE). These algorithms could potentially have better results than ICRL which independently trains each agent.

Additionally, our decision to fix the position of the ball and the goal in AntSoccerV1 and AntSoccerV2 (in comparison to the randomized placement in the AntBall environment) likely limits the agent's ability to generalize its behavior to be able to score with different goal and ball locations. Future work could analyze the affect of the randomization of the target and ball location and accordingly modify the RL algorithms to adapt to these changes.

A further optimization that could be made is a more thorough design of the reward function for applying PPO. In our implementations we experimented with only a subset of the large number of possible ways to model rewarding an agent for getting closer to scoring a goal. If the PPO reward function is designed in more innovative or generalizable ways then the learned policies could yield significantly different behavior. Additionally, applying grid search for hyperparameter tuning could help improve the dense reward function by finding more effective coefficients for the different reward terms, potentially leading to better overall performance.

References

[1] Anonymous (2025). Goal-conditioned multi-agent cooperation with contrastive rl.

[2] Bortkiewicz, M., Pałucki, W., Myers, V., Dziarmaga, T., Arczewski, T., Łukasz Kuciński, and Eysenbach, B. (2025). Accelerating goal-conditioned rl algorithms and research.

[3] Freeman, C. D., Frey, E., Raichuk, A., Girgin, S., Mordatch, I., and Bachem, O. (2021). Brax – a differentiable physics engine for large scale rigid body simulation.

[4] Liu, S., Lever, G., Wang, Z., Merel, J., Eslami, S. M. A., Hennes, D., Czarnecki, W. M., Tassa, Y., Omidshafiei, S., Abdolmaleki, A., Siegel, N. Y., Hasenclever, L., Marris, L., Tunyasuvunakool, S., Song, H. F., Wulfmeier, M., Muller, P., Haarnoja, T., Tracey, B. D., Tuyls, K., Graepel, T., and Heess, N. (2021). From motor control to team play in simulated humanoid football.

[5] Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. (2013). Playing atari with deep reinforcement learning.

[6] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., and Hassabis, D. (2016). Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489.

[7] Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., Lillicrap, T., Simonyan, K., and Hassabis, D. (2017). Mastering chess and shogi by self-play with a general reinforcement learning algorithm.