Continuous Mean-Zero Disagreement-Regularized Imitation Learning (CMZ-DRIL)

Noah Ford
noah.ford@jhuapl.edu
Johns Hopkins University
Applied Physics Laboratory
Laurel, MD, USA

Ryan W. Gardner
ryan.gardner@jhuapl.edu
Johns Hopkins University
Applied Physics Laboratory
Laurel, MD, USA

Austin Juhl
austin.juhl@jhuapl.edu
Johns Hopkins University
Applied Physics Laboratory
Laurel, MD, USA

Nathan Larson
nathan.larson@jhuapl.edu
Johns Hopkins University
Applied Physics Laboratory
Laurel, MD, USA

Abstract: Machine-learning paradigms such as imitation learning and reinforcement learning can generate highly performant agents in a variety of complex environments. However, commonly used methods require large quantities of data and/or a known reward function. Without extensive data, agents may find themselves outside of their training distribution, which could lead to unsafe behavior. This paper presents a method called Continuous Mean-Zero Disagreement-Regularized Imitation Learning (CMZ-DRIL) that employs a novel reward structure to improve the performance of imitation-learning agents that have access to only a handful of expert demonstrations. CMZ-DRIL uses reinforcement learning to minimize uncertainty among an ensemble of agents trained to model the expert demonstrations. This method does not use any environment-specific rewards, but creates a continuous and mean-zero reward function from the action disagreement of the agent ensemble, aiming to train the agent to return to states of higher certainty or those that resemble the training distribution. As demonstrated in a waypoint-navigation environment and in two MuJoCo environments, CMZ-DRIL can generate more performant agents than previous approaches in several key metrics.

Keywords: imitation learning, robust, uncertainty

1 Introduction

In control problems, machine-learning methods attempt to use environmental data to teach an agent to safely and accurately perform a task. Reinforcement Learning (RL) and Imitation Learning (IL) are the two most common types of machine-learning algorithms for these settings. Both types of methods typically require a large amount of environmental interaction or expert data, without which the resulting agent can exhibit emergent, and potentially unsafe, behavior in unseen states.

Reinforcement learning trains an agent toward acting optimally in an environment given a reward structure. RL has demonstrated remarkable success in numerous domains including board games [1], physical control [2], autonomous driving [3], and video games [4, 5]. However, RL also has challenges. First, it requires a reward function, which measures the quality of every achieved state, but such a function might not always be available or easy to construct. Second, RL methods are

Workshop on Safe and Robust Robot Learning for Operation in the Real World (SAFE-ROL) at CoRL 2024.

typically slow to converge, particularly in cases where a reward is not reached until after many specific steps, and require a large number of environmental interactions in the training process.

Imitation learning trains an agent to imitate the actions of a set of expert demonstrations instead of relying on a reward function. Imitation is among the most common ways humans learn to perform tasks. However, while humans learn quite well from a small number of demonstrations or a single demonstration, machine imitation-learning algorithms [6, 7, 8] are brittle and typically require a large number of examples to begin to learn to successfully perform a task. Without demonstrations that cover the entire state space, IL can have compounding error that can lead to catastrophic failure.

It is often difficult to collect large quantities of expert data in target environments. Some environments may have few demonstrations while other environments are too complex to reasonably represent all possible states. Because of the difficulties in collecting sufficient data, most agents are trained on incomplete data. To mitigate risk, imitation learning should teach an agent to generalize to unseen scenarios and states.

Inverse Reinforcement Learning (IRL) is one method for training agents to generalize to unseen states. By learning the reward of an expert agent, the imitator agent is able to predict how the expert would act in unseen states to maximize its reward [9, 10, 11]. While IRL provides a strong foundation for improving agent generalization, these methods tend to be difficult to train, often requiring extensive parameter tuning in new environments.

This paper presents a robust imitation learning algorithm called Continuous Mean-Zero Disagreement-Regularized Imitation Learning (CMZ-DRIL). CMZ-DRIL leverages ensemble-based uncertainty quantification to construct a training reward that intuitively aims to bring the agent to or keep the agent in states of maximal action-choice certainty. The agent ensemble is trained on a set of expert trajectories. CMZ-DRIL uses the ensemble uncertainty to construct a continuous reward function, achieving a mean of zero by subtracting the observed exponential average within the reward. This reward is optimized using RL. In contrast to the continuous CMZ-DRIL reward, the original DRIL [12] uses a clipped uncertainty cost that is a step function with 2 possible outputs. We hypothesize that the smooth reward improves the RL's ability to find the disagreement-minimization direction. CMZ-DRIL exhibits enhanced task performance using limited expert demonstrations and bypasses the need to identify an appropriate value of a critical threshold variable for each environment. This advancement is highlighted by comparing the performance of CMZ-DRIL agents to that of agents trained with Behavioral Cloning (BC) [13] and DRIL.

We evaluate CMZ-DRIL in three environments: two OpenAI Gym MuJoCo environments, Half Cheetah and Hopper, and a simple path planning environment, PyUXV [14]. Experimental results indicate that CMZ-DRIL can significantly improve performance of the imitator compared to previous key imitation learning algorithms BC and DRIL. CMZ-DRIL can close around half of the performance gap between BC and training with the true environmental reward. CMZ-DRIL achieves this performance by leveraging only expert trajectories and environmental interaction, and does not require additional task information, like a reward function, or environmental information. CMZ-DRIL or similar methods could help improve the safety of autonomous agents by reducing an agent's uncertainty during operation, encouraging the agent to stay away from unknown states that could lead to significant failures.

2 Related Work

Many algorithms have been proposed to improve the ability of a learning agent to generalize to unseen states. An ability to generalize can improve an agent's performance without requiring copious amounts of training data.

Several methods have been developed to use an explicit reward function to improve agents' ability to generalize when learning from demonstration. Deep Q-learning from Demonstrations (DQfD) [15] and Deep Deterministic Policy Gradient from Demonstration (DDPGfD) [16] are imitation-learning variants of Deep Q-Networks (DQN) that include demonstration data in the replay buffer and gives priority to demonstration data. The main criticism of these methods is that the demonstration data is underutilized. Policy Optimization from Demonstration (POfD) [17] is a variant of the method Generative Adversarial Imitation Learning (GAIL) [7] that reduces the sensitivity of GAIL to the quantity and quality of training data. While POfD shows promising results, it relies on training an adversarial network in parallel with the policy which can make the algorithm unstable and difficult to tune. Additionally, these method rely on a known reward function, which may not be available in certain environments.

IRL methods provide a natural framework for teaching agents to generalize to unseen states. Many traditional methods in IRL, such as Maximum Entropy IRL ([18]), are able to effectively learn the expert's reward structure, but require an explicit environment-transitional model. However, many environments are only known through sequential interactions such as interacting with a simulation. Adversarial methods, such as Adversarial Inverse Reinforcement Learning ([9]), learn a reward structure by discriminating between the expert's actions and the imitator's actions. While adversarial methods can learn accurate reward structures in many environments, they can be difficult to train, particularly in complex environments. More recent methods in Inverse Q Learning, like [10] and [19] can learn more stably by directly optimizing the Q function. These methods demonstrate highly performant agents on MuJoCo environments with only a single trajectory. Other methods, such as [11], also use direct optimization, but use a behavior-cloning loss to learn the reward.

Other algorithms take a different approach to the generalization problem by encouraging the imitator to stay close to states that have a sufficient number of expert demonstrations. By encouraging the imitator to stay near states with expert data, the imitator can more reliably predict the expert's actions and achieve better performance.

Recently, [20] introduced a method that uses methods from Lyapunov stability and density estimation to improve the ability of the agent stay within the training data's distribution. Additionally, the authors provide guarantees of the agent staying in distribution during the entirety of its trajectory. While this method shows promising results, it is possible that agent uncertainty does not directly correspond to data density. We are interested in directly using agent uncertainty instead of data density. We use agent uncertainty as a soft constraint, guiding the agent back to regions of higher certainty.

In this path of explicit uncertainty quantification, Brantley *et al.* introduced DRIL [12] which first trains an ensemble of imitators on the available expert data using BC [13] and then train a second agent using BC as well as RL with a reward that encourages the agent to stay in states with low disagreement between the ensemble agents. The idea is that the agent ensemble will agree well in parts of the state space where good expert data is available, so the new agent will be trained to remain in areas where it can most confidently model the expert data available. DRIL can offer improved performance in many Atari environments when limited demonstrations are available.

Like DRIL, CMZ-DRIL also quantifies uncertainty, or disagreement, as the standard deviation between an ensemble of imitators trained on expert data. Unlike DRIL, CMZ-DRIL uses a continuous reward function that includes an exponential average of past disagreement so the mean reward is approximately zero and does not require the user to identify an environment-specific threshold that defines positive or negative disagreement. DRIL rewards the agent either $+1$ or -1 if the ensemble standard deviation is below or above a user-defined threshold respectively. It is possible that this binary reward makes it difficult for the learning agent to progress gradually toward behaviors with lower disagreement in many settings. This reward may also be most frequently positive or most frequently negative for a given threshold, training environment, and data set, which can incentivize the agent to prolong episodes in potentially undesirable ways (if positive) or end them quickly (if negative).

3 Approach

CMZ-DRIL trains an agent using both BC and RL. For RL, CMZ-DRIL provides a continuous, mean-zero reward based on the standard deviation of an imitator ensemble.

In the course of developing CMZ-DRIL, a reward was tested that penalized proportionally to the ensemble's uncertainty level without providing any positive reward. While this reward improved the agent's performance in some environments, the negative value of the reward encouraged the agent to prematurely end episodes within the environment. For example, in tests performed during this study within the CartPole environment, the agent would quickly leave the environment's operational bounds to avoid accruing negative reward. To mitigate this unintended behavior, CMZ-DRIL was developed to provide a reward that averages to 0 over time.

CMZ-DRIL uses the reward $r_i = -\alpha(u_i - \bar{u}_i)$, where u_i is the ensemble standard deviation at time step i, and $\bar{u}_i = \gamma\bar{u}_{i-1} + (1-\gamma)u_{i-1}$ is the exponential average of the uncertainty experienced by the agent. Here, $\alpha > 0$ and $0 < \gamma < 1$ are constants. In experiments, the constants $\alpha = 10$, $\gamma = 0.99$, and $\bar{u}_0 = 0$ performed well. By using the standard deviation directly within the reward, uncertainty changes are gradual. We hypothesize that this semi-smooth reward can provide the learning agent a clearer sense of progress, thus allowing it to descend the uncertainty gradient. Additionally, by subtracting the exponential average within the reward calculation, CMZ-DRIL reduces the incentive to end the episode in a way that would cause the agent's performance to deviate from the expert's.

CMZ-DRIL trains an imitator agent in two steps: First, CMZ-DRIL uses the same expert data both to pretrain the imitator and to train the ensemble for uncertainty quantification. Then, CMZ-DRIL employs Proximal Policy Optimization (PPO) ([2]) to train the imitator based on the ensemble standard deviation. After each PPO step, the imitator is trained for an additional epoch on the expert data, using the Negative Log Likelihood (NLL), so that the current policy continues to choose the same actions as the expert in regions of high expert-data concentration.

Algorithm 1 CMZ-DRIL

```
Require: $\{O_e, A_e\}$, Expert observation-action pairs
  Pretrain:
  Train agent with $\{O_e, A_e\}$ using NLL
  Train ensemble with $\{O_e, A_e\}$
  Train:
  while Policy is not converged do
  Collect trajectory rollouts:
      $r_i = -\alpha(u_i - \bar{u}_i))$
      $\bar{u}_{i+1} = \gamma\bar{u}_i + (1-\gamma)u_i$
  Update agent with PPO
  Train agent with $\{O_e, A_e\}$ using NLL
  end while
```

Method	PyUXV	Half Cheetah	Hopper
Behavioral Cloning	-52.6 (45.0)	708 (162)	704(248)
DRIL Reward	Not Tested	277 (60.9)	244 (23.0)
CMZ-DRIL	-34.8 (29.4)	1690 (360)	1140 (514)

Table 1: Mean reward (and standard deviation) for the last 100 training epochs of each environment.

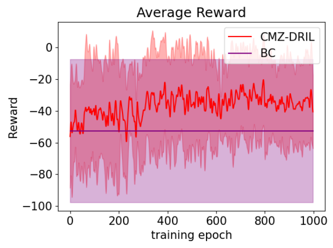
4 Experiments and Results

4.1 Experiments

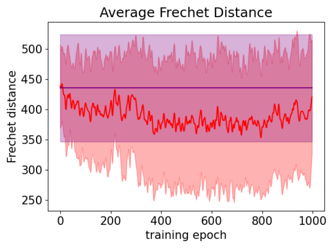
Three environments, each with an expert agent, were used to test the performance of CMZ-DRIL.

4.1.1 Environments

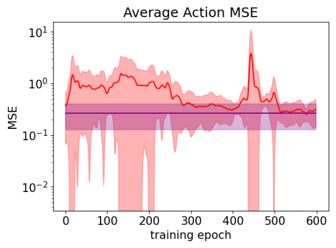
PyUXV PyUXV is a waypoint-navigation and obstacle-avoidance environment. The observation space includes lidar readings as well as position and goal information ([14]). The agent uses continuous heading and throttle control to navigate in a 2D plane to the waypoint while avoiding a



(a) Reward

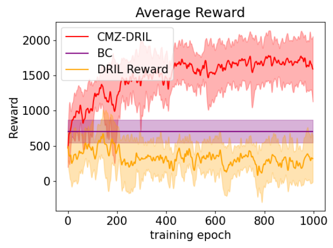


(b) Frechet Distance

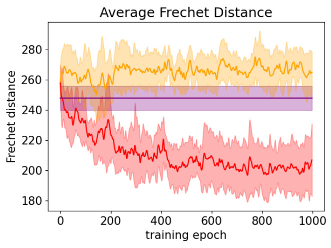


(c) Action MSE

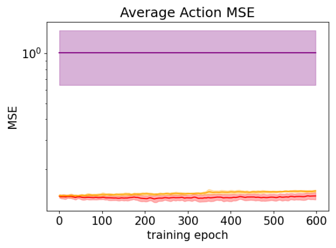
PyUXV



(d) Reward

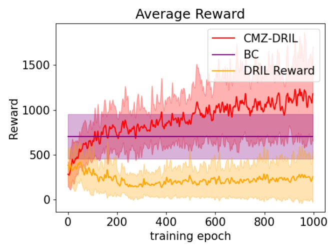


(e) Frechet Distance

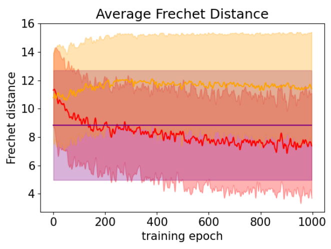


(f) Action MSE

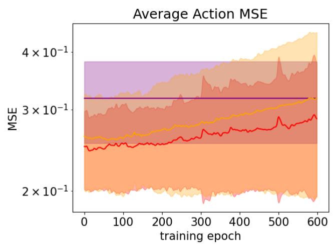
Half Cheetah



(g) Reward



(h) Frechet Distance



(i) Action MSE

Hopper

Figure 1: Reward, Frechet Distance, and MSE comparisons between CMZ-DRIL, BC, and DRIL

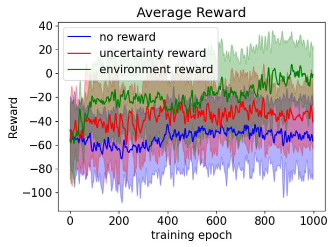
randomly-initialized field of obstacles. The expert for PyUXV is a reactive agent that minimizes its distance to the goal and avoids obstacles by following a tangent line to the obstacle.

MuJoCo Half Cheetah MuJoCo Half Cheetah (version 1) is a two-dimensional robot that controls a set of 8 joints to move itself forward. The goal in this environment is to run forward as fast as possible. The expert for Half Cheetah was generated using Soft Actor-Critic (SAC) from the implementation by [21]. The trained expert has a cumulative reward of 5551 on the training episodes.

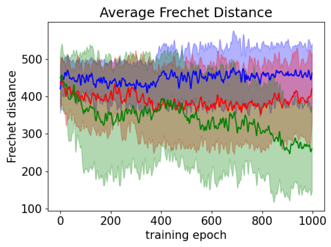
MuJoCo Hopper MuJoCo Hopper (version 1) is a two-dimensional leg-like robot with controllable joints. The goal in this environment is to hop forward as fast as possible. The expert for Hopper was generated using SAC from the implementation by [21]. The trained expert has a cumulative reward of 2863 on the training episodes.

4.1.2 Expert Demonstrations

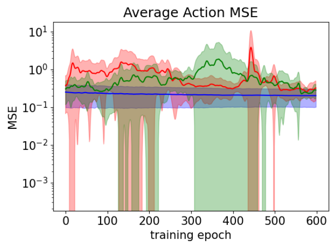
The set of expert demonstrations for one of our trials, i.e. the training data, includes 5 episodes for PyUXV, which consist of about 300 time-steps per training set, and 1 episode for Half Cheetah and Hopper each with around 1000 steps.



(a) Reward

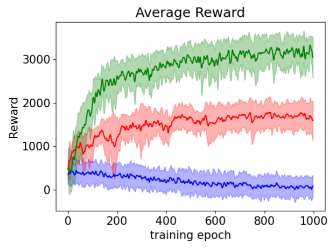


(b) Frechet Distance

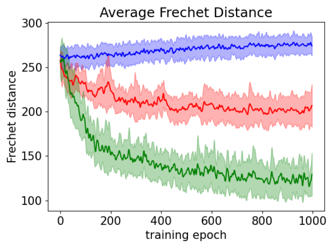


(c) Action MSE

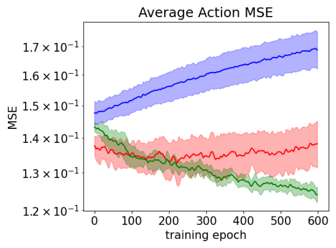
PyUXV



(d) Reward

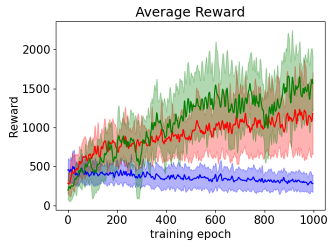


(e) Frechet Distance

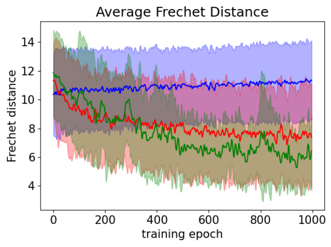


(f) Action MSE

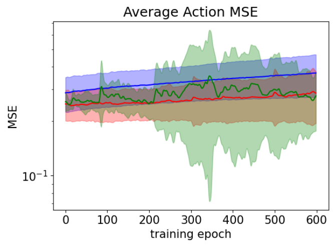
Half Cheetah



(g) Reward



(h) Frechet Distance



(i) Action MSE

Hopper

Figure 2: Comparison of agent performance with CMZ-DRIL reward, true environment reward, and no reward

4.1.3 Trials

We performed 5 training-and-evaluation trials on the MuJoCo environments for BC, CMZ-DRIL, and an implementation of DRIL that matches our implementation of CMZ-DRIL but with the DRIL reward and threshold parameters from the original DRIL paper [12]. We further compared CMZ-DRIL to BC on PyUXV although we did not include DRIL. We did not have the DRIL threshold parameter for that environment. Each trial used different training data, evaluation data, and network seeds. The same training and evaluation data were used for BC, DRIL, and CMZ-DRIL. We used 5 evaluation episodes for each experiment.

4.2 Results

Table 1 reports rewards achieved by BC, DRIL, and CMZ-DRIL. Figure 1 reports training results. We smooth the epoch-series data using a Gaussian filter with $\sigma = 2$ to better visualize the pattern.

Through training an imitator to minimize uncertainty during an episode, CMZ-DRIL achieved significant improvement on environmental reward and a lower Frechet distance compared to BC. In contrast, trials using the DRIL reward did not exhibit significant improvement. Additionally, the original DRIL method did not show improvement in Half Cheetah and Hopper performance compared to BC within the HalfCheetahBulletEnv-v0 and HopperBulletEnv-v0 environments as reported by [12].

Other than in the Half Cheetah environment, the Mean Squared Error (MSE) achieved by CMZ-DRIL is not significantly better than BC. In fact, the MSE does not appear to be correlated with the reward or Frechet distance metrics during the training process. It is possible that the reward for returning to known state space alters individual actions from what the expert would have done, leading to a higher MSE, even though the overall performance is more similar to the expert's.

Since CMZ-DRIL agents have improved performance compared to the BC agents, the contribution of ensemble-based reward to performance was also explored. Trials using the ensemble-based reward were compared to trials with no extra reward as well as trials in which the agent is trained with the true environmental reward. The trials that train with the true environmental reward represent an approximate upper bound on reward performance. Results are visualized in Figure 2. The epoch-series data is smoothed using a Gaussian filter with $\sigma = 2$ to better visualize the pattern. CMZ-DRIL agents performed better in reward, Frechet distance, and MSE than those trained with no additional reward. As expected, the CMZ-DRIL agents performed worse than agents trained with the true environmental reward. The environmental reward encodes additional task information into the training process, and is able to achieve a higher performance compared to methods that only use expert data. However, CMZ-DRIL was able to close the performance gap between agents trained with no reward and agents trained with the environmental reward by about 50%. The continuous, mean-zero reward provided by CMZ-DRIL reliably improves agent performance in the three environments tested.

5 Conclusions

CMZ-DRIL can train uncertainty-aware imitators that noticeably outperform BC in a waypoint-navigation environment as well as in two MuJoCo environments with RL-based experts. Additionally, this paper demonstrates the improved performance of agents trained with the CMZ-DRIL reward as opposed to agents trained with no reward. CMZ-DRIL demonstrates the power of staying within the distribution of the training data for machine-learning agents. Imitation-learning through simple supervised learning can lead to imitators with paths that leave regions of high data concentration, putting the imitator in states where it has not learned to perform safely or well. Methods that encourage agents to stay or return to regions of high data concentration have the potential to greatly improve safety and performance of the agents, particularly over longer time horizons.

CMZ-DRIL could provide a useful paradigm for training agents both during and after the agent's primary training process. If coupled with an exploratory learning method, the CMZ-DRIL method could help the agent refine its performance as it slowly explores other regions of the statespace. Along with other fine-tuning methods used to train foundation models, CMZ-DRIL could provide a framework to refine an agent's performance after training, helping to reduce or mitigate failure modes including harmful behavior. As fine-tuning becomes a more important component of the overall training process, recovery methods could become an important tool to help stabilize performance and safety.

Acknowledgments

This research is in part funded by the Test Resource Management Center (TRMC) and Test Evaluation/Science & Technology (T&E/S&T) Program under contract W900KK-19-C-004. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Test Resource Management Center (TRMC) or Test Evaluation/Science & Technology (T&E/S&T) Program.

References

[1] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T. Lillicrap, K. Simonyan, and D. Hassabis. A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science*, 362(6419): 1140–1144, 2018.

[2] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

[3] B. R. Kiran, I. Sobh, V. Talpaert, P. Mannion, A. A. A. Sallab, S. Yogamani, and P. Pérez. Deep reinforcement learning for autonomous driving: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 23(6), 2022.

[4] OpenAI, :, C. Berner, G. Brockman, B. Chan, V. Cheung, P. Dębiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse, R. Józefowicz, S. Gray, C. Olsson, J. Pachocki, M. Petrov, H. P. d. O. Pinto, J. Raiman, T. Salimans, J. Schlatter, J. Schneider, S. Sidor, I. Sutskever, J. Tang, F. Wolski, and S. Zhang. Dota 2 with large scale deep reinforcement learning, 2019. URL https://arxiv.org/abs/1912.06680. https://arxiv.org/abs/1912.06680.

[5] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev, J. Oh, D. Horgan, M. Kroiss, I. Danihelka, A. Huang, L. Sifre, T. Cai, J. P. Agapiou, M. Jaderberg, A. S. Vezhnevets, R. Leblond, T. Pohlen, V. Dalibard, D. Budden, Y. Sulsky, J. Molloy, T. L. Paine, C. Gulcehre, Z. Wang, T. Pfaff, Y. Wu, R. Ring, D. Yogatama, D. Wünsch, K. McKinney, O. Smith, T. Schaul, T. Lillicrap, K. Kavukcuoglu, D. Hassabis, C. Apps, and D. Silver. Grandmaster level in Starcraft II using multi-agent reinforcement learning. *Nature*, 575:350–354, 2019. URL http://dx.doi.org/10.1038/nature16961.

[6] M. Bain and C. Sammut. A framework for behavioural cloning. *Machine Intelligence*, 103(15), 1995.

[7] J. Ho and S. Ermon. Generative adversarial imitation learning. In *Neural Information Processing Systems (NeurIPS)*, 2016.

[8] F. Torabi, G. Warnell, and P. Stone. Behavioral cloning from observation. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2018.

[9] J. Fu, K. Luo, and S. Levine. Learning robust rewards with adversarial inverse reinforcement learning, 2018. URL https://arxiv.org/abs/1710.11248.

[10] D. Garg, S. Chakraborty, C. Cundy, J. Song, M. Geist, and S. Ermon. Iq-learn: Inverse soft-q learning for imitation, 2022. URL https://arxiv.org/abs/2106.12142.

[11] A. Szot, A. Zhang, D. Batra, Z. Kira, and F. Meier. Bc-irl: Learning generalizable reward functions from demonstrations. *arXiv preprint arXiv:2303.16194*, 2023.

[12] K. Brantley, M. Henaff, and W. Sun. Disagreement-regularized imitation learning. In *International Conference on Learning Representations (ICLR)*, 2020.

[13] M. Bain and C. Sammut. A framework for behavioural cloning. In *Machine Intelligence 15*, 1995. URL https://api.semanticscholar.org/CorpusID:10738655.

[14] C. A. Lowman, J. S. McClellan, and G. E. Mullins. Imitation learning with approximated behavior cloning loss. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8921–8927, 2021. doi:10.1109/IROS51168.2021.9636797.

[15] T. Hester, M. Vecerik, O. Pietquin, M. Lanctot, T. Schaul, B. Piot, D. Horgan, J. Quan, A. Sendonaris, G. Dulac-Arnold, I. Osband, J. Agapiou, J. Z. Leibo, and A. Gruslys. Deep q-learning from demonstrations, 2017.

[16] M. Vecerík, T. Hester, J. Scholz, F. Wang, O. Pietquin, B. Piot, N. M. O. Heess, T. Rothörl, T. Lampe, and M. A. Riedmiller. Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards. *ArXiv*, abs/1707.08817, 2017. URL https://api.semanticscholar.org/CorpusID:23192910.

[17] B. Kang, Z. Jie, and J. Feng. Policy optimization with demonstrations. In J. Dy and A. Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2469–2478. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/kang18a.html.

[18] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey. Maximum entropy inverse reinforcement learning. In D. Fox and C. P. Gomes, editors, *AAAI*, pages 1433–1438. AAAI Press, 2008. ISBN 978-1-57735-368-3. URL http://dblp.uni-trier.de/db/conf/aaai/aaai2008.html#ZiebartMBD08.

[19] F. Al-Hafez, D. Tateo, O. Arenz, G. Zhao, and J. Peters. Ls-iq: Implicit reward regularization for inverse reinforcement learning, 2023. URL https://arxiv.org/abs/2303.00599.

[20] K. Kang, P. Gradu, J. Choi, M. Janner, C. Tomlin, and S. Levine. Lyapunov density models: Constraining distribution shift in learning-based control, 2022. URL https://arxiv.org/abs/2206.10524.

[21] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22 (268):1–8, 2021. URL http://jmlr.org/papers/v22/20-1364.html.