Model Predictive Adversarial Imitation Learning for Planning from Observation

Tyler Han[*] Yanda Bao Bhaumik Mehta Gabriel Guo Anubhav Vishwakarma

Emily Kang Sanghun Jung Rosario Scalise Jason Zhou Bryan Xu Byron Boots

University of Washington

Abstract: Humans can often perform a new task after observing a few demonstrations by inferring the underlying intent. For robots, recovering the intent of the demonstrator through a learned reward function can enable more efficient, interpretable, and robust imitation through planning. A common paradigm for learning how to plan-from-demonstration involves first solving for a reward via Inverse Reinforcement Learning (IRL) and then deploying it via Model Predictive Control (MPC). In this work, we unify these two procedures by introducing planning-based Adversarial Imitation Learning, which simultaneously learns a reward and improves a planning-based agent through experience while using observation-only demonstrations. We study advantages of planning-based AIL in generalization, interpretability, robustness, and sample efficiency through experiments in simulated control tasks and real-world navigation from few- and single-demonstration.

Keywords: Imitation Learning, Model Predictive Control, Inverse Reinforcement Learning

1 Introduction

Inverse Reinforcement Learning (IRL) offers a principled approach to imitation learning by inferring the underlying intent, or reward function, that explains expert behavior. A fundamental advantage of IRL is that this reward is often readily generalizable beyond the support of the demonstration data, enabling the discovery of new policies through interaction and plans through self-prediction. Especially when demonstrations are sparse, ambiguous, or suboptimal, IRL's interpretability is particularly well-suited for domains where understanding preferences and ensuring reliable planning are essential, such as routing on Google Maps [1], socially aware navigation [2], and autonomous driving [3].

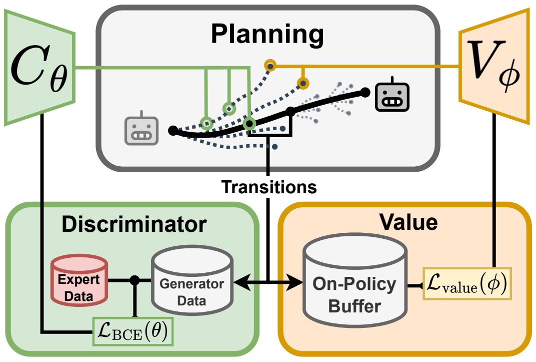

Figure 1: Model Predictive Adversarial Imitation Learning (MPAIL) embeds an MPC agent within Adversarial Imitation Learning (AIL) for learning costs from observation and interaction. On deployment, the MPC agent performs online optimization by evaluating short model rollouts (dotted lines) with the learned costs and value networks.

For real-time systems, learned IRL and Inverse Optimal Control (IOC) rewards are typically deployed via Model Predictive Control

[*]Corresponding author: than123@uw.edu

(MPC) [4, 5, 6, 7, 8, 9, 10]. Here, the offline IRL algorithm iteratively solves a Reinforcement Learning (RL) problem in an inner loop, guided by the current reward estimate. An outer loop then updates this reward to minimize the discrepancy between the agent's and the expert's behavior. Once training is complete, the resulting reward is integrated with MPC for real-time planning and control.

Adversarial Imitation Learning (AIL) has made significant improvements over IRL in algorithmic complexity and sample efficiency [11, 12]. However, the reliance on an RL policy in AIL methods complicates their use in applications with safety constraints [4, 6, 7]. Further limited by partial observability, these deployments will often prioritize planning using a model for the sake of real-time performance, trustworthiness, and interpretability [13, 14, 15].

In this work, we derive *planning-based* AIL, yielding key benefits:

1. **Planning-from-Observation (PfO).** Towards interpretable yet scalable imitation learning, a predictive model precludes the need for expert action data and enables access to the agent's optimization landscape. This grants crucial insight and steerability into the agent's decision making process even as it learns from ambiguous expert data. *We further show that this improves on out-of-distribution generalization, robustness, and sample efficiency when compared to policy-based AIL. We also demonstrate how policy-based AIL is fundamentally limited by the absence of reward deployment.*
2. **Unification of IRL and MPC.** Otherwise considered independent training and deployment procedures, planning-based AIL allows for end-to-end interactive learning of the entire planner. Critical online settings (e.g., dynamics, preferences, control constraints) can thus be brought into training while enabling experience-based reasoning beyond the planning horizon, which we demonstrate in this work. *We also find this induces a more effective adversarial dynamic than policy-based generators when learning from partial observations in the real world.*

To our knowledge, this work presents the first end-to-end planning-from-observation (PfO) framework, extending PfO to continuous spaces and interactive learning. By choosing Model Predictive Path Integral control (MPPI) [16] as the embedded planner, we further gain theoretical perspective on planning-based AIL and its relationship to the seminal GAIL objective [11, 17]. Thus, we name this learning algorithm: Model Predictive Adversarial Imitation Learning (MPAIL {*impale*}).

2 Related Work

IRL-MPC. High-dimensional continuous control applications often require an online planner for real-time control, trustworthiness, safety, or additional constraints. When using IRL to learn a reward, online deployments of these reward functions tend to rely on an independent online MPC procedure. To enable learning local costmaps across perception and control for off-road navigation, Lee et al. [4] and Triest et al. [6] similarly propose solving the forward RL problem by using MPPI but deploy the learned reward on a different configuration more suitable for real-time planning and control. This reward deployment framework of *IRL-then-MPC* is currently the dominant approach for planning in high-dimensional continuous control tasks from demonstration [4, 5, 6, 7, 8, 9, 10].

Model-Based IRL and Planning-Based RL. The proposed framework, MPAIL, might naively be categorized as a model-based AIL approach. Various other works have also explored model-based AIL [18, 3, 19]. However, scope is directed at training stabilization and policy optimization rather than examining planning with learned reward. When the reward is known, as in RL, planning-based algorithms have demonstrated considerable improvement in simulation benchmarks over existing state-of-the-art RL algorithms through developments such as: online trajectory optimization, value bootstrapping, latent state planning, policy-like or learned sampling priors, and much more [20, 21, 22, 23]. This work's implementation of MPAIL performs online trajectory optimization and value bootstrapping. This work *does not* implement latent state planning nor a policy-based prior to better isolate our investigations in interpretability and planning [24, 19].

3 Model Predictive Adversarial Imitation Learning

3.1 The POMDP Setting and the Model Predictive Agent

We adopt the Partially Observable Markov Decision Process (POMDP) to best consider highly desirable applications of IRL in which partial observability and model-based planning play crucial roles. In an unknown world state $s_w \in \mathcal{S}_w$, the agent makes an observation $o \sim p(o|s_w)$. From a history of observations $\mathbf{o}_{0:t}$, the Agent perceives its *state* $s_t \sim p(s|\mathbf{o}_{0:t})$. *Actions* $a \in \mathcal{A}$ and states $s \in \mathcal{S}$ together allow the agent to self-predict forward in time using its *predictive model* $f : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{S}$. Note that these definitions crucially suggest the partial observability of s due to the implicit dependency on the observation history $\mathbf{o}_{0:t}$ through the agent's perception (e.g. mapping [25]). However, partial observations in $\mathcal{S}$ are desirably used for demonstrations to perform IRL and AIL, as full observation history would quickly become intractable [6]. The planner itself is a model predictive agent. It is capable of performing *model rollouts* $\tau_t^{(H)} = \{s_{t'}, a_{t'}\}_{t'=t}^{t+H}$ such that $s_{t'+1} = f(s_{t'}, a_{t'})$. Each rollout is then mapped to a cost $C(\tau_t)$. Finally, the agent's objective is to create an H-step action sequence $\mathbf{a}_{t:t+H}$, or *plan*, that best minimizes its corresponding trajectory cost.

3.2 Adversarial Imitation Learning from Observation

IRL algorithms aim to learn a cost function that minimizes the cost of expert trajectories while maximizing the cost of trajectories induced by other policies [17, 11]. As the problem is ill-posed and many costs can correspond to a given set of demonstrations, the principle of maximum entropy is imposed to obtain a uniquely optimal cost. It can be shown that the entropy maximizing distribution is a Boltzmann distribution [26]. Towards scalable learning-from-observation (LfO), we further consider demonstration data in which only states are available, as actions can be challenging or impossible to obtain [27]. In this context, the state-only IRL from observation problem can be formulated by costing state-transitions $c(s, s')$ rather than state-actions $c(s, a)$ as in [17]:

$$\text{IRLfO}_\psi(\pi_E) = \underset{c \in \mathbb{R}^{\mathcal{S} \times \mathcal{S}}}{\operatorname{argmax}} -\psi(c) + \left(\min_{\pi \in \Pi} -\lambda \mathbb{H}(\pi) + \mathbb{E}_\pi[c(s, s')] \right) - \mathbb{E}_{\pi_E}[c(s, s')], \tag{1}$$

where $\psi(c)$ is a convex cost regularizer, π_E is the expert policy, $\mathbb{H}(\cdot)$ is the entropy, and Π is a family of policies.

As shown in [11, 17], this objective can be shown to be dual to the Adversarial Imitation Learning (AIL) objective under a specific choice of cost regularizer ψ,

$$\min_{\pi \in \Pi} \max_{D \in [0,1]^{\mathcal{S} \times \mathcal{S}}} \mathbb{E}_\pi[\log(D(s, s'))] + \mathbb{E}_{\pi_E}[\log(1 - D(s, s'))] - \lambda \mathbb{H}(\pi), \tag{2}$$

where $D(\cdot)$ is the discriminator function. The exact form of D has consequences on the policy objective and differs by AIL algorithm. Now equipped with the optimization objective, we continue in our derivation of planning-based AIL by choosing the form of the policy class and reward function.

3.3 Choosing a Policy and Reward

To reiterate, we set our sights on the AIL objective (Equation (2)) which aims to simultaneously learn reward and policy from demonstration. However, the formulation remains intimately connected with policy optimization through the assumption of an RL procedure. Towards planning-based optimization, we proceed by modifying the RL formulation in Equation (1) as described in [11, 24]. Similar to [28], we replace the entropy loss $-\lambda \mathbb{H}(\pi)$ with a Kullback-Leibeler (KL) divergence constraint on the previous policy $\overline{\pi}$:

$$\min_{\pi \in \Pi} \mathbb{E}_\pi[c(s, s')] + \beta \, \mathbb{KL}(\pi \,||\, \overline{\pi}). \tag{3}$$

Note that this incorporates prior information about the policy (i.e. previous plans) while seeking the next maximum entropy policy. Specifically, as shown in Section B.1, the closed form solution to

Equation (3) is $\pi^*(a|s) \propto \overline{\pi}(a|s)e^{\frac{-1}{\beta}\overline{c}(s,a)}$, where $\overline{c}(s,a) = \sum_{s' \in \mathcal{S}} \mathcal{T}(S_{t+1} = s'|S_t = s)c(s,s')$ and $\mathcal{T}(S_{t+1} = s'|S_t = s)$ denotes the transition probability from state s to s'.

We then observe that a choice of planner satisfies the RL objective as in Equation (3). By choosing Model Predictive Path Integral (MPPI) as the planner, as proven in Section B.1, we solve an equivalent problem provided the MDP is uniformly ergodic. Namely, MPPI solves for a KL-constrained cost-minimizer over trajectories [28]:

$$\min_{\pi \in \Pi} \mathbb{E}_{\tau \sim \pi} \left[C(\tau) + \beta \, \mathbb{KL}(\pi(\tau) \, || \, \overline{\pi}(\tau)) \right] \tag{4}$$

where $C(\tau)$ is the discounted cost of a trajectory and $\mathbb{KL}(\pi(\tau)||\overline{\pi}(\tau))$ is the discounted KL divergence over a trajectory.

For practical reasons, model rollouts are often limited to some timestep length, H. However, this can often result in myopic plans or limit applications to short-horizon tasks [21]. To resolve this, infinite-horizon MPPI introduces a terminal cost-to-go function to be evaluated on the final states in the rollouts [21, 29, 30, 22]. This can be done by utilizing a learned value function $V_\phi : \mathcal{S} \to \mathbb{R}$ that estimates the expected return G_t of a state s_t as $G_t = \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + ...|S_t = s_t]$, where $R_{t+1} = R(s_t, s_{t+1})$. The result of MPPI's approximately global policy optimization at each timestep is what is referred to as the *MPPI policy*, π_{MPPI}. Section B.1 proves how this formulation can be equivalent to the entropy-regularized RL objective, while also in the observation-only setting. Pseudocode for the MPPI procedure can be found in Algorithm 2 as well as for its adaptation as an RL policy in Algorithm 3. Figure 2 illustrates the policy.

In short, our chosen AIL agent, infinite-horizon MPPI, can be viewed as an approximately global optimizer for an entropy-regularized policy at a given state. Borrowing from the online learning perspective [31], these optimizations occur online rather than offline as part of, for instance, the canonical actor-critic update.

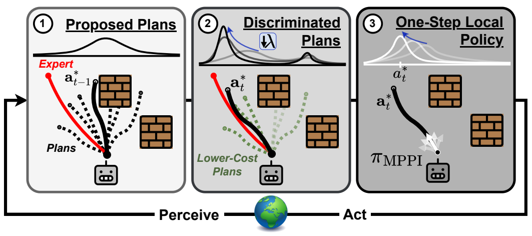

Figure 2: Illustration of π_{MPPI} in MPAIL. (1) A set of action sequences (plans) are sampled and rolled out. (2) Plans are costed according to the discriminator, shifting the distribution towards the expert. Temperature λ decreases during training, narrowing the optimized distribution in subsequent episodes. (3) The policy π_{MPPI} is the result of a Gaussian fit to the optimized plans and their respective first actions.

Provided the agent (policy), we now proceed with selecting its objective. Recent work has shown many potential choices of valid policy objectives, each with various empirical tradeoffs [32]. We found the reward as defined in Adversarial Inverse Reinforcement Learning (AIRL) [24] to be most stable when combined with the value function when applied to infinite-horizon MPPI. In the state-only setting, the policy objective becomes $r(s,s') = \log(D(s,s')) - \log(1 - D(s,s'))$ and the discriminator $D(s,s') = \sigma \circ d_\theta(s,s')$. Simply put, the reward is the logit of the discriminator $r(s,s') = d_\theta(s,s')$.

In summary, our choice of π_{MPPI} and $r(s,s')$ yields the MPAIL framework. As proven in Section B.2, MPAIL is indeed an AIL algorithm in the sense that it minimizes divergence from the expert policy. The procedure (Algorithm 1) itself closely resembles the original GAIL procedure. However, upon updating the value network, MPAIL does not require a policy update thereafter. We also find in practice that a temperature decay can be helpful for preventing local minima, especially in the case of online model learning. We leave a theoretical justification for this choice for future work. An overview of the training procedure can be found in Figure 1. A discussion of further meaningful implementation details, like spectral normalization, can be found in Section C. Though, these modifications are kept to a minimum towards our analysis of π_{MPPI} in AIL.

Algorithm 1 Model Predictive Adversarial Imitation Learning

```
Require: Expert state-transitions $\mathcal{D}_E = \{(s, s')\}$
Require: Maximum-Entropy Planner $\pi_{\text{MPPI}}$, Discriminator $D_\theta$, Value $V_\phi$
1: while not converged do
2:     Collect transitions $(s, s', r_\theta(\cdot)) \in d^\pi$ by running $\pi_{\text{MPPI}}$ (Alg. 3) in the environment
3:     Update Discriminator parameters using $\theta$:
           $\nabla_\theta \mathbb{E}_{s,s' \sim d^\pi}[log(D_\theta(s, s'))] + \nabla_\theta \mathbb{E}_{s,s' \sim d^{\pi_E}}[log(1 - D_\theta(s, s'))]$    (5)
4:     Update Value parameters $\phi$ using estimated returns:
           $\nabla_\phi \mathbb{E}_{s \sim d^\pi}[(G_t - V_\phi(s))^2]$    (6)
5: end while
```

4 Experimental Results

Questions. Without a policy, vanilla MPPI possesses no "memory" about actions taken in previous episodes, save for those implied through the value $V_\phi(s)$. In addition, MPPI can be viewed as an online, zeroth-order optimization as opposed to an offline, first-order optimization as in policy gradient methods. These novelties raise a critically practical question about planning-based AIL; are policies learned online through planning sufficient as adversarially generative policies? In our experiments, we find that MPAIL indeed trains an effective imitator, provoking our follow-up questions:

Q1 What is the advantage of deploying an AIL planner over an AIL policy?

Q2 How does MPAIL help enable real-world planning capabilities from observation?

Q3 How does MPAIL compare to existing AIL algorithms?

Hyperparameter settings are kept consistent across all experiments. Exact values and other implementation details such as regularization and computation are reported and discussed in Section C.

4.1 Simulated Navigation Task

For simulated evaluation, we design a navigation task with a 10-DoF vehicle. Reward is proportional to the negative squared distance to $(10, 10)$. Initial poses ($t = 0$) are within 1 m of $(0, 0)$. The state is 12-dimensions: position, orientation, linear velocity, and angular velocity. Actions include target velocity and steering angle. MPAIL plans using an approximate prior model, the Kinematic Bicycle Model [33]. This approximate model is not tuned to the agent dynamics. For instance, slipping and suspension dynamics occur in simulation but are unmodeled [33].

An expert is trained on this environment using PPO [34] and is used for expert data. At convergence, the optimal policy occasionally circles near the goal instead of stopping on it precisely (see Figure 9). While the total return between these two behaviors are nearly identical, we choose to use the circling demonstrations as expert data, because it is a more challenging behavior to imitate. This is corroborated by Orsini et al. [32], who stress that demonstrator suboptimality and multimodality is a critical component in algorithm evaluation towards practical AIL from human data.

The expert's circling behavior presents itself as a challenging "distractor mode". In training, the policy may begin to only circle around the origin. If the AIL algorithm is not able to sufficiently explore, training collapses on this mode where the policy continuously circles the origin, unable to return to the expert distribution which requires the circling behavior to occur around the goal. For instance, we find that AIRL is unable to successfully learn both behaviors likely due to the instability introduced by its logit shift [32].

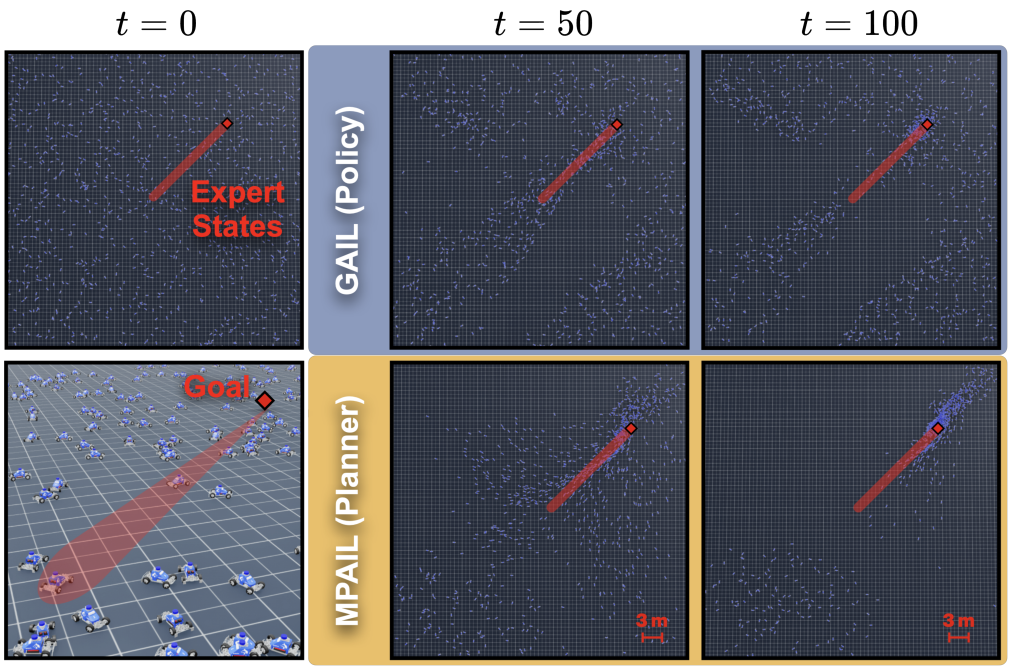

Figure 3: **Comparison of policy-based and planning-based AIL in Out-of-Distribution (OOD) states.** Agents trained on the navigation task (Section 4.1) are placed uniformly with random orientation between a 40×40 m box centered on $(0, 0)$. The policy and planner are run for 100 timesteps in the environment. Data support of the expert exists mainly between $(0, 0)$ and $(10, 10)$[*]. Quantitative evaluation of this experiment can be found in Figure 4.

4.2 Out-of-Distribution Recovery Through Planning – Q1

When deploying learning-based methods to the real-world, reliable performance in out-of-distribution (OOD) states are of critical importance, especially in imitation learning when expert data can be extremely sparse. We show that planning-based AIL (MPAIL) improves generalization capabilities when OOD. In this experiment, we use the simulated navigation environment but *expand the region of uniformly distributed initial positions and orientations from a 1×1 square to a large 40×40 m square around the origin* (Figure 3). The policy-based approach is represented by GAIL, as AIRL does not meaningfully converge in the navigation task (Section 4.1). Only four expert demonstrations are used in training.

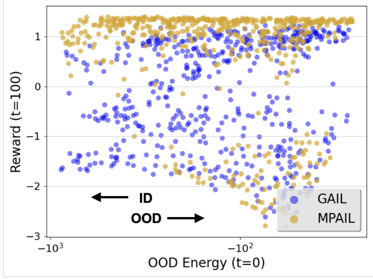

Figure 4: **Quantitative OOD Comparison.** Agent initial poses vary from In-distribution (ID) to OOD relative to the expert data and are plotted with their final reward after 100 timesteps. Metric from [35] (see Section D.2).

We find that planning-based AIL generalizes to significantly more states than policy-based AIL when outside the support of expert data. In this experiment, the planner's horizon is a maximum of 3 meters. As a result, the task horizon may be up to 15 times longer than the planning horizon. Evidently, a planner could not navigate to the goal if the learned optimization landscape (induced by cost c_θ and value V_ϕ) did not also generalize to OOD states. These results suggest a fundamental limitation of current AIL approaches and their single policy solution. The trained reward and value are inefficiently underutilized in policy-based AIL and not utilized at all on deployment. By contrast, *MPAIL re-introduces the reward and value online to solve for new policies each moment in time*. These results illustrate that generalization in AIL is substantially improved through reward deployment in addition to reward learning.

[*]Note that the state space is 12-dimensional; expert data support is extremely sparse in this environment.

4.3 Real-Sim-Real Navigation from a Single Demonstration – Q2

Real-world evaluation of AIL is currently challenging. RL-like interaction efficiency renders training in simulation more practical than in the real-world [36], but demonstrations must realistically still be from the real-world. Nonetheless, it is imperative to evaluate AIL methods on real-world sub-optimal data and hardware since results may diverge significantly from ideal settings and simulation [32, 37].

Our hardware experiment evaluates GAIL, IRL-MPC, and MPAIL through Real-to-Sim-to-Real: 1) a *single* partially observable (position and body-centric velocity) demonstration is collected from the real-world, 2) the method is trained using interactions from simulation, finally 3) the method is deployed zero-shot to the real-world for evaluation. This experiment uses a small-scale RC car platform with an NVIDIA Jetson Orin NX [38]. For IRL-MPC, the reward and value is trained through GAIL, which must be hand-tuned before deployment on MPPI.

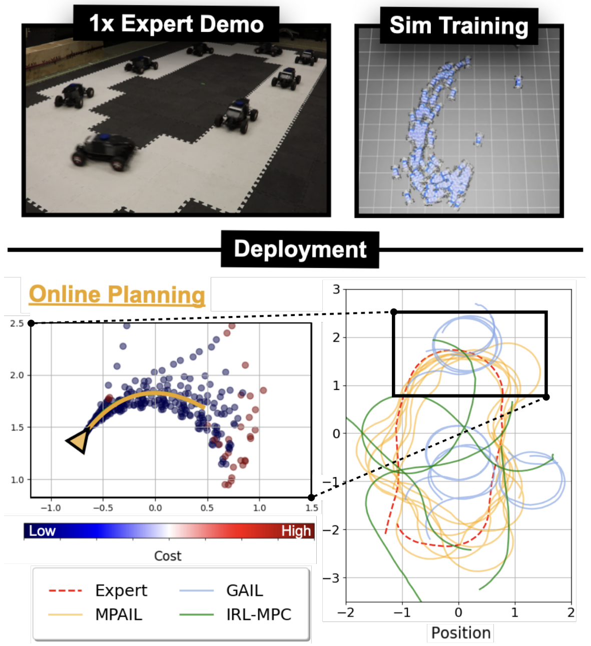

Figure 5: **Real-Sim-Real Experiment. Bottom Left (MPAIL).** Real-time (20 Hz) parallel model rollouts and costing are visualized while the robot navigates through the turn. Current optimal plan for the next 1 second in gold. **Bottom Right.** Trajectories performed by MPAIL, GAIL, and IRL-MPC (see Table 1 for evaluation).

We find that MPAIL is able to qualitatively reproduce the expert trajectory with an average Relative Cross-Track-Error (CTE [39]) of 0.17 m while traveling an average of 0.3 m/s slower. In addition, Figure 5 illustrates a key advantage of planning-based AIL. By granting access to the agent's optimization landscape, MPAIL significantly improves on the interpretability of agents trained through ambiguous and complex human demonstration data when compared to black-box policies. Note the lower costing of on-track trajectories and final plan.

GAIL does not reliably converge to the expert even in training. During deployment, GAIL's policy consistently veers off-path or collapses into driving in a circle. Various starting configurations were attempted without success. While literature on the evaluation of AIL methods in the real-world are sparse, we find that AIL policies can be extremely poor performing in the real-world, as corroborated by [19]. A more detailed discussion is provided in Section D.1.

IRL-MPC acts as a middle-ground between MPAIL and GAIL; the learned reward and value are exactly the same as GAIL's and thus differs by the deployment of the reward through planning. IRL-MPC's improvements over GAIL provides evidence that: (i) model-based planning can grant robustness to a model-free reward and, (ii) despite GAIL's poor performance, the learned reward was still meaningfully discriminative and suggests a failure of the policy to optimize this reward. On the other hand, IRL-MPC diverges from MPAIL by mainly learned reward and value. As a result, we find that online policy optimization through π_{MPPI} induces a more competitive adversarial dynamic than offline policy optimization as in actor-critic RL. In this case, the end-to-end inclusion of the planner enables training the reward and value to completion.

	CTE (m)		
	Max	Mean	Average Speed (m/s)
Expert	-	-	1.0
GAIL	1.29	0.56	0.37
IRL-MPC	1.28	0.37	0.30
MPAIL	0.76	0.17	0.70

Table 1: **Evaluation of Real Experiment.** Relative Cross-Track Error (CTE) and speed are computed over the best five laps.

4.4 Benchmarking – Q3

While we have shown theoretical justification for the classification of MPAIL as an AIL algorithm, we perform additional benchmarking experiments for empirical validation. We train GAIL, AIRL, and MPAIL on the navigation task and the cartpole task across varying quantities of expert demonstrations and random seeds as done in [11, 40]. While MPAIL uses an approximate prior model for the navigation task, we choose to learn a model during training of the cartpole task to demonstrate the generality of MPC and support future work on additional tasks. This is represented by the label, *MPAIL (OM)*, indicating that there is a fully **O**nline **M**odel. Implementation details of the learned model can be found in Section D.3.

On the navigation task, MPAIL reaches optimality in less than half the number of interactions when compared to GAIL. We also observe MPAIL to train more stably than GAIL on this task. AIRL struggles to learn from the multi-modal data (see Section 4.1). This navigation benchmark helps support MPAIL's characterization as a model-based AIL algorithm as it is more sample-efficient when provided a prior model. On the cartpole task, expert demonstration data results in optimal-but-sparse state visitation and, equivalently, AIL reward signal. It is likely that, due to online dynamics model learning, MPAIL requires more exploratory interactions to combat a large local minima induced and reinforced by sparse discriminator reward, model bias, and task dynamics. Asymptotically, MPAIL attains comparable performance while maintaining the benefits of model-based planning, such as interpretability and robustness.

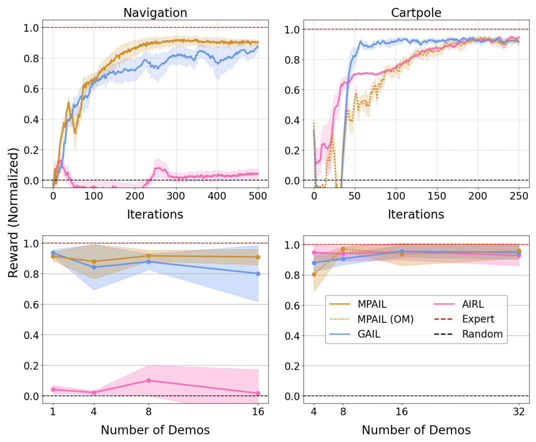

Figure 6: **Benchmarking Results.** Top row rewards are computed across all demonstration quantities and seeds. Bottom row rewards are the average of the final 10 episodes computed across seeds.

5 Conclusion

In this work, we address the Planning-from-Observation (PfO) setting by introducing planning-based AIL and, our algorithmic implementation, MPAIL. By comparing to existing AIL algorithms in out-of-distribution settings, we reveal how reward deployment—not only reward learning as in current AIL—is critical towards generalizable imitation learning. MPAIL addresses this by re-introducing the learned reward online to solve for new policies at each time step. This perspective unifies IRL and MPC under a single interactive learning algorithm, enabling continual improvement of a planning-based agent with a handful of demonstrations. From a single partially observable demonstration in real-world navigation, we find that MPAIL is the only successful imitator when compared to policy-based AIL and IRL-MPC. Towards safe and interpretable robot learning, MPAIL employs representations (i.e. model, reward, value) which grant access to the agent's optimization landscape and thus decision-making process.

MPAIL is derived from, and naturally admits, abstractions from Model Predictive Control, model-based RL, and imitation learning. Thus, its open-source implementation aims to reflect this and offers common ground for instantiating the many possible extensions to adjacent work through these connections: off-policy value estimation for improved sample efficiency or offline learning [40], policy-like proposal distributions and latent dynamics for scaling MPPI [20], model-free and model-based reward blending for alleviating model bias [21], diffusion-inspired MPPI for improved online optimization [41], and much more. We envision that this work can provide a theoretically and empirically justified foundation for future work at the intersection of MPC, RL, and Imitation Learning.

6 Limitations

As noted in Section 3, the implications of the temperature decay remains to be investigated. We suspect that there may be a more adaptable approach towards scheduling or balancing its effects similar to well-studied entropy-regularized RL methods.

Vanilla MPPI is also known to struggle with high-dimensional tasks [41]. We direct the interested reader to Section D.5 to view results and discussion on the higher-dimensional ($\mathcal{S} \subseteq \mathbb{R}^{60}$; $\mathcal{A} \subseteq \mathbb{R}^{8}$) Ant task. However, vanilla MPPI still demonstrates promise towards scaling MPAIL through learned walking behavior. We believe that a biased sampling distribution, integration of existing MPPI optimization improvements [41], and latent state planning [20] offer promising solutions for scaling MPAIL to higher dimensions.

Consistent with recent findings on the limited reliability of simulation-based AIL when applied to human demonstrations [32], our experiments further highlight that real-world deployment of AIL would benefit from additional investigation [19]. In particular, choosing appropriate evaluation models can be cumbersome, especially in settings without clear performance metrics [42].

As AIL is fundamentally an interactive learning algorithm, MPAIL benefits from real-world RL approaches which seek to make training in the real-world practical. This would help alleviate losses in performance due to reward entanglement with environment dynamics by precluding the Real-Sim-Real alternative.

In Figure 3, stark OOD configurations around $(-15, -15)$ remain challenging for cost and value. In our real-world experiments, un-trained regions of cost or reward are still capable of leading the agent astray. Albeit, this occurs at far lesser rates and impact than policy networks due to MPPI's connections to optimal control and disturbance rejection. Further regularization or structural insights into training the discriminator are promising directions towards further robustifying these networks.

Acknowledgments

TH is supported by the NSF GRFP under Grant No. DGE 2140004.

References

[1] M. Barnes, M. Abueg, O. F. Lange, M. Deeds, J. Trader, D. Molitor, M. Wulfmeier, and S. O'Banion. Massively Scalable Inverse Reinforcement Learning in Google Maps. Oct. 2023. URL `https://openreview.net/forum?id=z3L59iGALM`.

[2] H. Kretzschmar, M. Spies, C. Sprunk, and W. Burgard. Socially compliant mobile robot navigation via inverse reinforcement learning. *The International Journal of Robotics Research*, 35(11):1289–1307, Sept. 2016. ISSN 0278-3649. doi:10.1177/0278364915619772. URL `https://doi.org/10.1177/0278364915619772`. Publisher: SAGE Publications Ltd STM.

[3] E. Bronstein, M. Palatucci, D. Notz, B. White, A. Kuefler, Y. Lu, S. Paul, P. Nikdel, P. Mougin, H. Chen, J. Fu, A. Abrams, P. Shah, E. Racah, B. Frenkel, S. Whiteson, and D. Anguelov. Hierarchical Model-Based Imitation Learning for Planning in Autonomous Driving, Oct. 2022. URL `http://arxiv.org/abs/2210.09539`. arXiv:2210.09539 [cs].

[4] K. Lee, D. Isele, E. A. Theodorou, and S. Bae. Spatiotemporal Costmap Inference for MPC Via Deep Inverse Reinforcement Learning. *IEEE Robotics and Automation Letters*, 7(2):3194–3201, Apr. 2022. ISSN 2377-3766. doi:10.1109/LRA.2022.3146635. URL `https://ieeexplore.ieee.org/abstract/document/9695208?casa_token=QJYQX8gpAREAAAAA:ViJkcxuQmUk7q-rlEXUndFekQplZ61lk5SDAa1Bwb_XvoHzz5XF7xHasSJB_lzmBjd3tdnjd-w`. Conference Name: IEEE Robotics and Automation Letters.

[5] S. Rosbach, V. James, S. Großjohann, S. Homoceanu, and S. Roth. Driving with Style: Inverse Reinforcement Learning in General-Purpose Planning for Automated Driving. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2658–2665, Nov. 2019. doi:10.1109/IROS40897.2019.8968205. URL `https://ieeexplore.ieee.org/document/8968205/?arnumber=8968205`. ISSN: 2153-0866.

[6] S. Triest, M. G. Castro, P. Maheshwari, M. Sivaprakasam, W. Wang, and S. Scherer. Learning Risk-Aware Costmaps via Inverse Reinforcement Learning for Off-Road Navigation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 924–930, May 2023. doi:10.1109/ICRA48891.2023.10161268. URL `https://ieeexplore.ieee.org/document/10161268/`.

[7] N. Das, S. Bechtle, T. Davchev, D. Jayaraman, A. Rai, and F. Meier. Model-Based Inverse Reinforcement Learning from Visual Demonstrations. In *Proceedings of the 2020 Conference on Robot Learning*, pages 1930–1942. PMLR, Oct. 2021. URL `https://proceedings.mlr.press/v155/das21a.html`. ISSN: 2640-3498.

[8] K. Lee, B. Vlahov, J. Gibson, J. M. Rehg, and E. A. Theodorou. Approximate Inverse Reinforcement Learning from Vision-based Imitation Learning. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10793–10799, May 2021. doi:10.1109/ICRA48506.2021.9560916. URL `https://ieeexplore.ieee.org/document/9560916/`. ISSN: 2577-087X.

[9] M. Kuderer, S. Gulati, and W. Burgard. Learning driving styles for autonomous vehicles from demonstration. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2641–2646, May 2015. doi:10.1109/ICRA.2015.7139555. URL `https://ieeexplore.ieee.org/document/7139555/`. ISSN: 1050-4729.

[10] K. Lee, D. Isele, E. A. Theodorou, and S. Bae. Risk-sensitive MPCs with Deep Distributional Inverse RL for Autonomous Driving. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7635–7642, Oct. 2022. doi:10.1109/IROS47612.2022.9981223. URL `https://ieeexplore.ieee.org/document/9981223/`. ISSN: 2153-0866.

[11] J. Ho and S. Ermon. Generative Adversarial Imitation Learning. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. URL `https://proceedings.neurips.cc/paper_files/paper/2016/hash/cc7e2b878868cbae992d1fb743995d8f-Abstract.html`.

[12] N. Baram, O. Anschel, I. Caspi, and S. Mannor. End-to-End Differentiable Adversarial Imitation Learning. In *Proceedings of the 34th International Conference on Machine Learning*, pages 390–399. PMLR, July 2017. URL `https://proceedings.mlr.press/v70/baram17a.html`. ISSN: 2640-3498.

[13] T. Han, A. Liu, A. Li, A. Spitzer, G. Shi, and B. Boots. Model Predictive Control for Aggressive Driving Over Uneven Terrain. In *Robotics: Science and Systems XX*. Robotics: Science and Systems Foundation, July 2024. ISBN 9798990284807. doi:10.15607/RSS.2024.XX.022. URL `http://www.roboticsproceedings.org/rss20/p022.pdf`.

[14] C. Katrakazas, M. Quddus, W.-H. Chen, and L. Deka. Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions. *Transportation Research Part C: Emerging Technologies*, 60:416–442, Nov. 2015. ISSN 0968-090X. doi:10.1016/j.trc.2015.09.011. URL `https://www.sciencedirect.com/science/article/pii/S0968090X15003447`.

[15] S. Choudhury, M. Bhardwaj, S. Arora, A. Kapoor, G. Ranade, S. Scherer, and D. Dey. Data-driven planning via imitation learning. *The International Journal of Robotics Research*, 37(13-14):1632–1672, Dec. 2018. ISSN 0278-3649. doi:10.1177/0278364918781001. URL `https://doi.org/10.1177/0278364918781001`. Publisher: SAGE Publications Ltd STM.

[16] G. Williams, N. Wagener, B. Goldfain, P. Drews, J. M. Rehg, B. Boots, and E. A. Theodorou. Information theoretic MPC for model-based reinforcement learning. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1714–1721, May 2017. doi:10.1109/ICRA.2017.7989202. URL `https://ieeexplore.ieee.org/document/7989202`.

[17] F. Torabi, G. Warnell, and P. Stone. Generative Adversarial Imitation from Observation, June 2019. URL `http://arxiv.org/abs/1807.06158`. arXiv:1807.06158 [cs].

[18] N. Baram, O. Anschel, and S. Mannor. Model-based Adversarial Imitation Learning, Dec. 2016. URL `http://arxiv.org/abs/1612.02179`. arXiv:1612.02179 [stat].

[19] J. Sun, L. Yu, P. Dong, B. Lu, and B. Zhou. Adversarial Inverse Reinforcement Learning With Self-Attention Dynamics Model. *IEEE Robotics and Automation Letters*, 6(2):1880–1886, Apr. 2021. ISSN 2377-3766. doi:10.1109/LRA.2021.3061397. URL `https://ieeexplore.ieee.org/document/9361118`.

[20] N. Hansen, H. Su, and X. Wang. TD-MPC2: Scalable, Robust World Models for Continuous Control, Mar. 2024. URL `http://arxiv.org/abs/2310.16828`. arXiv:2310.16828 [cs].

[21] M. Bhardwaj, S. Choudhury, and B. Boots. Blending MPC & Value Function Approximation for Efficient Reinforcement Learning, Apr. 2021. URL `http://arxiv.org/abs/2012.05909`. arXiv:2012.05909 [cs].

[22] K. Lowrey, A. Rajeswaran, S. Kakade, E. Todorov, and I. Mordatch. Plan Online, Learn Offline: Efficient Learning and Exploration via Model-Based Control, Jan. 2019. URL `http://arxiv.org/abs/1811.01848`. arXiv:1811.01848 [cs].

[23] N. Jawale, B. Boots, B. Sundaralingam, and M. Bhardwaj. Dynamic Non-Prehensile Object Transport via Model-Predictive Reinforcement Learning, Nov. 2024. URL http://arxiv.org/abs/2412.00086. arXiv:2412.00086 [cs].

[24] J. Fu, K. Luo, and S. Levine. Learning Robust Rewards with Adversarial Inverse Reinforcement Learning, Aug. 2018. URL http://arxiv.org/abs/1710.11248. arXiv:1710.11248 [cs].

[25] S. Jung, J. Lee, X. Meng, B. Boots, and A. Lambert. V-STRONG: Visual Self-Supervised Traversability Learning for Off-road Navigation, Mar. 2024. URL http://arxiv.org/abs/2312.16016. arXiv:2312.16016 [cs].

[26] B. D. Ziebart, A. Maas, J. A. Bagnell, and A. K. Dey. Maximum Entropy Inverse Reinforcement Learning.

[27] F. Torabi, G. Warnell, and P. Stone. Recent Advances in Imitation Learning from Observation, June 2019. URL http://arxiv.org/abs/1905.13566. arXiv:1905.13566 [cs].

[28] M. Bhardwaj, A. Handa, D. Fox, and B. Boots. Information Theoretic Model Predictive Q-Learning.

[29] M. Zhong, M. Johnson, Y. Tassa, T. Erez, and E. Todorov. Value function approximation and model predictive control. In *2013 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*, pages 100–107, Apr. 2013. doi:10.1109/ADPRL.2013.6614995. URL https://ieeexplore.ieee.org/document/6614995/?arnumber=6614995. ISSN: 2325-1867.

[30] N. Hatch and B. Boots. The Value of Planning for Infinite-Horizon Model Predictive Control. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7372–7378, May 2021. doi:10.1109/ICRA48506.2021.9561718. URL https://ieeexplore.ieee.org/document/9561718. ISSN: 2577-087X.

[31] N. Wagener, C.-A. Cheng, J. Sacks, and B. Boots. An Online Learning Approach to Model Predictive Control, Oct. 2019. URL http://arxiv.org/abs/1902.08967. arXiv:1902.08967 [cs].

[32] M. Orsini, A. Raichuk, L. Hussenot, D. Vincent, R. Dadashi, S. Girgin, M. Geist, O. Bachem, O. Pietquin, and M. Andrychowicz. What Matters for Adversarial Imitation Learning? In *Advances in Neural Information Processing Systems*, volume 34, pages 14656–14668. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/hash/7b647a7d88f4d6319bf0d600d168dbeb-Abstract.html.

[33] T. Han, S. Talia, R. Panicker, P. Shah, N. Jawale, and B. Boots. Dynamics Models in the Aggressive Off-Road Driving Regime, May 2024. URL http://arxiv.org/abs/2405.16487. arXiv:2405.16487 [cs].

[34] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal Policy Optimization Algorithms, Aug. 2017. URL http://arxiv.org/abs/1707.06347. arXiv:1707.06347 [cs].

[35] W. Liu, X. Wang, J. Owens, and Y. Li. Energy-based Out-of-distribution Detection. In *Advances in Neural Information Processing Systems*, volume 33, pages 21464–21475. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/hash/f5496252609c43eb8a3d147ab9b9c006-Abstract.html.

[36] L. Tai, J. Zhang, M. Liu, and W. Burgard. Socially Compliant Navigation Through Raw Depth Inputs with Generative Adversarial Imitation Learning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1111–1117, May 2018. doi:10.1109/ICRA.2018.8460968. URL https://ieeexplore.ieee.org/document/8460968/. ISSN: 2577-087X.

[37] Y. Tsurumine and T. Matsubara. Goal-aware generative adversarial imitation learning from imperfect demonstration for robotic cloth manipulation. *Robotics and Autonomous Systems*, 158:104264, Dec. 2022. ISSN 0921-8890. doi:10.1016/j.robot.2022.104264. URL https://www.sciencedirect.com/science/article/pii/S0921889022001543.

[38] S. S. Srinivasa, P. Lancaster, J. Michalove, M. Schmittle, C. Summers, M. Rockett, R. Scalise, J. R. Smith, S. Choudhury, C. Mavrogiannis, and F. Sadeghi. MuSHR: A Low-Cost, Open-Source Robotic Racecar for Education and Research, Dec. 2023. URL http://arxiv.org/abs/1908.08031. arXiv:1908.08031 [cs].

[39] J. D. Rounsaville, J. S. Dvorak, and T. S. Stombaugh. Methods for Calculating Relative Cross-Track Error for ASABE/ISO Standard 12188-2 from Discrete Measurements.

[40] I. Kostrikov, K. K. Agrawal, D. Dwibedi, S. Levine, and J. Tompson. Discriminator-Actor-Critic: Addressing Sample Inefficiency and Reward Bias in Adversarial Imitation Learning, Oct. 2018. URL http://arxiv.org/abs/1809.02925. arXiv:1809.02925 [cs].

[41] H. Xue, C. Pan, Z. Yi, G. Qu, and G. Shi. Full-Order Sampling-Based MPC for Torque-Level Locomotion Control via Diffusion-Style Annealing, Sept. 2024. URL http://arxiv.org/abs/2409.15610. arXiv:2409.15610 [cs].

[42] L. Hussenot, M. Andrychowicz, D. Vincent, R. Dadashi, A. Raichuk, S. Ramos, N. Momchev, S. Girgin, R. Marinier, L. Stafiniak, M. Orsini, O. Bachem, M. Geist, and O. Pietquin. Hyperparameter Selection for Imitation Learning. In *Proceedings of the 38th International Conference on Machine Learning*, pages 4511–4522. PMLR, July 2021. URL https://proceedings.mlr.press/v139/hussenot21a.html. ISSN: 2640-3498.

[43] S. K. S. Ghasemipour, R. Zemel, and S. Gu. A Divergence Minimization Perspective on Imitation Learning Methods. In *Proceedings of the Conference on Robot Learning*, pages 1259–1277. PMLR, May 2020. URL https://proceedings.mlr.press/v100/ghasemipour20a.html. ISSN: 2640-3498.

[44] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. URL https://proceedings.neurips.cc/paper_files/paper/2014/hash/f033ed80deb0234979a61f95710dbe25-Abstract.html.

[45] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida. Spectral Normalization for Generative Adversarial Networks, Feb. 2018. URL http://arxiv.org/abs/1802.05957. arXiv:1802.05957 [cs].

[46] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning, July 2019. URL http://arxiv.org/abs/1509.02971. arXiv:1509.02971 [cs].

[47] T. Luo, T. Pearce, H. Chen, J. Chen, and J. Zhu. C-GAIL: Stabilizing Generative Adversarial Imitation Learning with Control Theory, Oct. 2024. URL http://arxiv.org/abs/2402.16349. arXiv:2402.16349 [cs].

[48] J. Schulman, S. Levine, P. Moritz, M. I. Jordan, and P. Abbeel. Trust Region Policy Optimization, Apr. 2017. URL http://arxiv.org/abs/1502.05477. arXiv:1502.05477 [cs].

[49] J. Geldenbott and K. Leung. Legible and Proactive Robot Planning for Prosocial Human-Robot Interactions. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13397–13403, May 2024. doi:10.1109/ICRA57147.2024.10611294. URL https://ieeexplore.ieee.org/document/10611294/.

[50] N. Hansen, X. Wang, and H. Su. Temporal Difference Learning for Model Predictive Control, July 2022. URL http://arxiv.org/abs/2203.04955. arXiv:2203.04955 [cs].

[51] A. S. Morgan, D. Nandha, G. Chalvatzaki, C. D'Eramo, A. M. Dollar, and J. Peters. Model Predictive Actor-Critic: Accelerating Robot Skill Acquisition with Deep Reinforcement Learning. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6672–6678, May 2021. doi:10.1109/ICRA48506.2021.9561298. URL https://ieeexplore.ieee.org/document/9561298/?arnumber=9561298. ISSN: 2577-087X.

[52] C. Finn, S. Levine, and P. Abbeel. Guided Cost Learning: Deep Inverse Optimal Control via Policy Optimization. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 49–58. PMLR, June 2016. URL https://proceedings.mlr.press/v48/finn16.html. ISSN: 1938-7228.

[53] N. Rudin, D. Hoeller, P. Reist, and M. Hutter. Learning to Walk in Minutes Using Massively Parallel Deep Reinforcement Learning. In *Proceedings of the 5th Conference on Robot Learning*, pages 91–100. PMLR, Jan. 2022. URL https://proceedings.mlr.press/v164/rudin22a.html. ISSN: 2640-3498.

[54] M. Mittal, C. Yu, Q. Yu, J. Liu, N. Rudin, D. Hoeller, J. L. Yuan, R. Singh, Y. Guo, H. Mazhar, A. Mandlekar, B. Babich, G. State, M. Hutter, and A. Garg. Orbit: A Unified Simulation Framework for Interactive Robot Learning Environments. *IEEE Robotics and Automation Letters*, 8(6):3740–3747, June 2023. ISSN 2377-3766, 2377-3774. doi:10.1109/LRA.2023.3270034. URL http://arxiv.org/abs/2301.04195. arXiv:2301.04195 [cs].

[55] T. Han, S. Rajagopal, Y. Bao, S. Jung, S. Talia, G. Guo, B. Xu, B. Mehta, E. Romig, R. Scalise, and B. Boots. Wheeled Lab: Modern Sim2Real for Low-Cost, Open-Source Wheeled Robotics. In *Proceedings of the Conference on Robot Learning*, pages 1259–1277. PMLR, Sept. 2025.

[56] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library, Dec. 2019. URL http://arxiv.org/abs/1912.01703. arXiv:1912.01703 [cs].

[57] B. Vlahov, J. Gibson, M. Gandhi, and E. A. Theodorou. MPPI-Generic: A CUDA Library for Stochastic Trajectory Optimization, Mar. 2025. URL http://arxiv.org/abs/2409.07563. arXiv:2409.07563 [cs].

[58] B. Vlahov, J. Gibson, D. D. Fan, P. Spieler, A.-a. Agha-mohammadi, and E. A. Theodorou. Low Frequency Sampling in Model Predictive Path Integral Control. *IEEE Robotics and Automation Letters*, 9(5):4543–4550, May 2024. ISSN 2377-3766, 2377-3774. doi:10.1109/LRA.2024.3382530. URL http://arxiv.org/abs/2404.03094. arXiv:2404.03094 [cs].

[59] Y. Li, J. Song, and S. Ermon. InfoGAIL: Interpretable Imitation Learning from Visual Demonstrations, Nov. 2017. URL http://arxiv.org/abs/1703.08840. arXiv:1703.08840 [cs].

[60] J. Sacks and B. Boots. Learning Sampling Distributions for Model Predictive Control. In *Proceedings of The 6th Conference on Robot Learning*, pages 1733–1742. PMLR, Mar. 2023. URL https://proceedings.mlr.press/v205/sacks23a.html. ISSN: 2640-3498.

[61] J. Sacks, R. Rana, K. Huang, A. Spitzer, G. Shi, and B. Boots. Deep Model Predictive Optimization. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 16945–16953, May 2024. doi:10.1109/ICRA57147.2024.10611492. URL https://ieeexplore.ieee.org/document/10611492/.

A Infinite Horizon Model Predictive Path Integral

In this section we present the full algorithm in detail, including MPPI as described in [16]. Modifications to "conventional" MPPI for MPAIL are highlighted in blue. Where applicable, $(\mathbf{x})_i$ indicates the ith entry of $\mathbf{x}$ (in its first dimension, if $\mathbf{x}$ is a tensor).

Algorithm 2 MPPI

```
Require:
    Number of trajectories to sample $N$;
    Planning horizon $H$;
    Number of optimization iterations $J$
    Fixed action sampling variance $\Sigma$;
    Previous optimal plan $\mathbf{a}^*_{t-1} = \{(\mathbf{a}^*_{t-1})_{t'}\}^H_{t'=0}$;
    Current state $s_t$;
    Dynamics model $f_\psi(s, a)$
    Costs $c_\theta(s, s')$
    Value $V_\phi(s)$
 1: Procedure MPPI($s_t, \mathbf{a}^*_{t-1}$)
 2: $(\mathbf{a}_t)^0_i \leftarrow (\mathbf{a}^*_{t-1})_{i+1}$                ▷ Roll previous plan one timestep forward
 3: $(\mathbf{a}_t)^0_H \leftarrow 0$                ▷ Set sampling mean to 0 for last timestep
 4: for $j \leftarrow 0$ to $J - 1$ do
 5:     for $k \leftarrow 0$ to $N - 1$ do                ▷ Model rollouts and costing (parallelized)
 6:         $\tilde{s}^k_0 \leftarrow s_t$
 7:         for $t' \leftarrow 0$ to $H - 1$ do
 8:             $a^k_{t'} \sim \mathcal{N}((\mathbf{a}_t)_{t'}, \Sigma)$                ▷ Sample action at predicted state
 9:             $\tilde{s}^k_{t'+1} \leftarrow f_\psi(\tilde{s}^k_{t'}, a^k_{t'})$                ▷ Predict next state
10:             $c^k_{t'} \leftarrow c_\theta(s^k_{t'}, s^k_{t'+1})$                ▷ Compute state-transition costs
11:         end for
12:         $\mathcal{C}(\tau_k) \leftarrow -\eta^H V_\phi(\tilde{s}^k_H) + \sum_{t'=0}^{H-1} \eta^{t'} c^k_{t'}$                ▷ Total trajectory cost
13:     end for
14:     $\beta \leftarrow \min_k[\mathcal{C}(\tau_k)]$
15:     $\mathcal{Z} \leftarrow \sum_{k=1}^{n} \exp -\frac{1}{\lambda}\mathcal{C}(\tau_k)$
16:     for $k \leftarrow 0$ to $N - 1$ do                ▷ Weight using exponential negative cost
17:         $w(\tau_k) \leftarrow \frac{1}{\mathcal{Z}} \exp -\frac{1}{\lambda}\mathcal{C}(\tau_k)$
18:     end for
19:     for $t' \leftarrow 0$ to $H - 1$ do                ▷ Optimal plan from weighted-average actions
20:         $(\mathbf{a}^j_t)_{t'} \leftarrow \sum_{k=0}^{N-1} w(\tau_k) a^k_{t'}$
21:     end for
22: end for
23: for $i \leftarrow 0$ to $|\mathcal{A}|$ do                ▷ Compute optimized standard deviations for policy
24:     $(\boldsymbol{\sigma}_t)_i \leftarrow \sqrt{\sum_{k=0}^{N-1} w(\tau_k)[((\mathbf{a}^J_t)_0)_i - (a^k_0)_i]^2}$
25: end for
26: return $\mathbf{a}^J_t, \boldsymbol{\sigma}_t$
27: End Procedure
```

B Proofs

In Section 3, we introduce the replacement of the entropy loss in Equation (2) with a KL divergence loss. This replacement allows the MPPI planner, in place of a policy, to solve the required forward RL problem. Integrated with the AIL objective in Equation (2), we further show that this allows MPAIL to correctly recover the expert state occupancy distribution $\rho_E(s, s')$. In this section, we prove both of these claims.

Algorithm 3 π_{MPPI}

```
Require:
    Reward $r_\theta := -c_\theta$;
    Value $V_\phi$;
    $\text{MPPI}(s, \mathbf{a}) = \text{MPPI}(s, \mathbf{a}; N, H, J, \Sigma, f_\psi, c_\theta, V_\phi)$ (Algorithm 2);
    $T$ length of episode
 1: $\mathbf{a}_0^* \leftarrow 0$                                    ▷ Initialize optimal plan
 2: $\mathcal{B} \leftarrow \{\}$
 3: for $t \leftarrow 1$ to $T$ do
 4:     $s_t \sim \mathcal{T}(\cdot|s_{t-1}, a_{t-1})$                ▷ Step and perceive environment
 5:     $\mathbf{a}_t^*, \boldsymbol{\sigma}_t \leftarrow \text{MPPI}(s_t, \mathbf{a}_{t-1}^*)$
 6:     if Train then
 7:         $a_t \sim \mathcal{N}((\mathbf{a}_t^*)_0, I\boldsymbol{\sigma}_t)$
 8:     else if Deploy then
 9:         $a_t \leftarrow (\mathbf{a}_t^*)_0$
10:     end if
11:     $r_t \leftarrow r_\theta(s_t, s_{t+1})$                        ▷ Reward from discriminator
12:     $\mathcal{B} \leftarrow \mathcal{B} \cup (s_t, a_t, r_t, s_{t+1})$
13: end for
14: return $\mathcal{B}$
```

B.1 MPPI as a Policy

In this section we justify claims regarding MPPI in the forward RL problem. For completeness, we also verify that known results remain consistent with our state-only restriction.

Proposition B.1.1. *The closed form solution of Equation* (3),

$$\min_{\pi \in \Pi} \mathbb{E}_\pi[c(s, s')] + \beta\, \mathbb{KL}(\pi \,||\, \overline{\pi}), \tag{3}$$

is

$$\pi^*(a|s) \propto \overline{\pi}(a|s) e^{\frac{-1}{\beta}\overline{c}(s,a)} \quad \textit{where} \quad \overline{c}(s, a) = \sum_{s' \in \mathcal{S}} \mathcal{T}(S_{t+1} = s'|S_t = s) c(s, s') \tag{7}$$

Proof. We begin by noting that

$$\mathbb{E}_\pi[c(s, s')|S_t = s] = \sum_{a \in \mathcal{A}} \pi(a|s)\overline{c}(s, a) \tag{8}$$

where the weighted cost $\overline{c}(s, a)$ is defined as

$$\overline{c}(s, a) := \sum_{s' \in \mathcal{S}} \mathcal{T}(S_{t+1} = s'|S_t = s) c(s, s') \tag{9}$$

Then for a fixed state $s \in \mathcal{S}$, noting that the policy is normalized over actions $\sum_{a \in \mathcal{A}} \pi(a|s) = 1$, we may form the Lagrangian with respect to the objective in Equation (3) as:

$$\mathcal{L}(\pi, \beta, \lambda) = \sum_{a \in \mathcal{A}} \pi(a|s)\overline{c}(s, a) \; + \; \beta \sum_{a \in \mathcal{A}} \pi(a|s) \log \frac{\pi(a|s)}{\overline{\pi}(a|s)} \; + \; \lambda \sum_{a \in \mathcal{A}} \pi(a|s) \; - 1 \tag{10}$$

Taking the partial derivative with respect to $\pi(a|s)$ and setting to 0 we have

$$\frac{\partial \mathcal{L}}{\partial \pi(a|s)} = \overline{c}(s, a) + \beta \log \frac{\pi(a|s)}{\overline{\pi}(a|s)} + 1 + \lambda = 0 \tag{11}$$

Finally,

$$\beta \log \frac{\pi(a|s)}{\overline{\pi}(a|s)} = -\overline{c}(s, a) - 1 - \lambda \tag{12}$$

$$\pi(a|s) \propto \overline{\pi}(a|s) e^{-\frac{1}{\beta}(\overline{c}(s,a)+1+\lambda)} \tag{13}$$

$$\pi(a|s) \propto \overline{\pi}(a|s) e^{-\frac{1}{\beta}\overline{c}(s,a)}$$

□

Remark B.1.2. *Given a uniform policy prior, the KL Objective in Equation* (3),

$$\min_{\pi \in \Pi} \mathbb{E}_\pi[c(s,s')] + \beta\, \mathbb{KL}(\pi \,||\, \overline{\pi}), \tag{3}$$

is equivalent to the Entropy Objective,

$$\min_{\pi \in \Pi} \mathbb{E}_\pi[c(s,s')] - \lambda\, \mathbb{H}(\pi). \tag{14}$$

Proof. In order to prove this, it suffices to note that minimizing KL is equivalent to maximizing entropy:

$$\begin{aligned}
\mathbb{KL}(\pi||\overline{\pi}) &= \sum_{s\in\mathcal{S}} d^\pi(s) \sum_{a\in\mathcal{A}} \pi(a|s) \log \frac{\pi(a|s)}{\overline{\pi}(a|s)} && (15)\\
&= \sum_{s\in\mathcal{S}} d^\pi(s) \left[-\mathbb{H}(\pi(\cdot|s)) - \sum_{a\in\mathcal{A}} \pi(a|s) \log \overline{\pi}(a|s) \right] && (16)\\
&= -\sum_{s\in\mathcal{S}} d^\pi(s)\mathbb{H}(\pi(\cdot|s)) - \sum_{s\in\mathcal{S}}\sum_{a\in\mathcal{A}} \pi(a|s) \log \overline{\pi}(a|s) && (17)\\
&= -\mathbb{H}(\pi) - \sum_{s\in\mathcal{S}} \log k_s && (18)
\end{aligned}$$

where $k_s = \overline{\pi}(a|s)$ for any $a \in \mathcal{A}$. Note that the sum on the left collapses by definition and the inner sum on the right collapses since the probability of taking an action in any given state is 1. Finally, since all the k_s are constant, the second term on the right hand side is constant. Since both objectives differ by a constant, minimizing the KL is equivalent to maximizing the Entropy given a uniform policy prior. □

Proposition B.1.3. *Provided the MDP is uniformly ergodic, the MPPI objective in Equation* (4),

$$\min_{\pi \in \Pi} \; \mathbb{E}_{\tau\sim\pi}\left[C(\tau) + \beta\, \mathbb{KL}(\pi(\tau) \,||\, \overline{\pi}(\tau))\right] \tag{4}$$

is equivalent to the RL objective in Equation (3),

$$\min_{\pi \in \Pi} \mathbb{E}_\pi[c(s,s')] + \beta\, \mathbb{KL}(\pi \,||\, \overline{\pi}). \tag{3}$$

Proof. Before continuing, we verify that infinite horizon MPPI indeed predicts an infinite horizon estimate of the return. For simplicity, we momentarily revert to a reward only formulation, replacing the cost $c_\theta(s,s')$ with the reward $R_\theta(s,s')$ and the control discount η with γ. We proceed by expanding the return,

$$\begin{aligned}
\mathbb{E}_{\tau\sim\pi}[R(\tau)] &= \mathbb{E}_{\tau\sim\pi}[\gamma^H V_\phi(s_H) + \sum_{t=1}^{H-1} \gamma^t R(s_t, s_{t+1})] && (19)\\
&= \mathbb{E}_{\tau\sim\pi}[\mathbb{E}_\pi[\sum_{t=H}^{\infty} \gamma^t R(s_t, s_{t+1}) | S_H = s_H] + \sum_{t=1}^{H-1} \gamma^t R(s_t, s_{t+1})] && (20)\\
&= \mathbb{E}_{\tau\sim\pi}[\sum_{t=H}^{\infty} \gamma^t R(s_t, s_{t+1}) + \sum_{t=1}^{H-1} \gamma^t R(s_t, s_{t+1})] && (21)\\
&= \mathbb{E}_{\tau\sim\pi}[\sum_{t=1}^{\infty} \gamma^t R(s_t, s_{t+1})] && (22)
\end{aligned}$$

where we have made use of the definition of a value function V_ϕ (Equations 18 to 19) and the tower property of expectation (Equations 19 to 20).

Let $f(s,s') = c(s,s') + \beta \mathbb{KL}(\pi(\cdot|s)||\overline{\pi}(\cdot|s))$. For either objective to be valid the cost and KL Divergence would have to be bounded. Thus, we may safely assume that f is uniformly bounded $||f||_\infty \leq K$. Let $\delta_t = \mathbb{E}_{s_t,s_{t+1}\sim d^t}[f(s_t,s_{t+1})] - \mathbb{E}_{s,s'\sim d^\pi}[f(s,s')]$ be the error between estimates of the objective.

Since the MDP is uniformly ergodic, we may bound the rate of convergence of the state distribution at a time t, d^t to the stationary distribution d^π

$$\exists \lambda \in (0,1), M \in \mathbb{N} \text{ s.t } ||d^t - d^\pi||_{\text{TV}} \leq M\lambda^t \tag{23}$$

where $||\cdot||_{\text{TV}}$ is the total variation metric.

Continuing by bounding error δ_t,

$$|\delta_t| \leq ||f||_\infty ||d^t - d^\pi||_{\text{TV}} \leq KM\lambda^t \tag{24}$$

We then have that $|\sum_{t=0}^\infty \eta^t \delta_t| \leq KM \sum_{t=0}^\infty (\eta\lambda)^t = \frac{KM}{1-\eta\lambda} = C < \infty$.

We may now begin working with the MPPI Objective in Equation (4)

$$\begin{aligned}
&\mathbb{E}_{\tau\sim\pi}[C(\tau) + \beta\,\mathbb{KL}(\pi(\tau)\,||\,\overline{\pi}(\tau))] && (25)\\
&= \sum_{t=0}^\infty \eta^t\, \mathbb{E}_{s_t,s_{t+1}\sim d^t}[f(s_t,s_{t+1})] && (26)\\
&= \sum_{t=0}^\infty \eta^t \left[\mathbb{E}_{s,s'\sim d^\pi}[f(s,s')] + \delta_t\right] && (27)\\
&= \frac{1}{1-\eta}\mathbb{E}_{s,s'\sim d^\pi}[f(s,s')] + \sum_{t=0}^\infty \eta^t \delta_t && (28)
\end{aligned}$$

Note that the MPPI objective and Entropy Regularized RL objective differ by scaling and a bounded additive constant, independent of π. Thus, minimizing both objectives are equivalent. □

B.2 MPAIL as an Adversarial Imitation Learning Algorithm

In this section, we integrate findings from Section B.1 with the AIL objective to theoretically validate MPAIL as an AIL algorithm. Specifically, we observe that at optimality, we recover the log expert-policy transition density ratio, which in turns yields a maximum entropy policy on state-transitions. We then discuss the identifiability limits imposed by observing only (s,s') rather than (s,a,s'). Throughout this section we make use of the state-transition occupancy measure, defined as $\rho_\pi : \mathcal{S}\times\mathcal{S} \to \mathbb{R}$ where $\rho_\pi(s,s') = \sum_{t=1}^\infty \gamma^t \mathcal{T}(S_{t+1}=s', S_t = s\,|\,\pi)$ as in [17].

Proposition B.2.1. *The optimal reward is*

$$f_\theta^*(s,s') = \log\left(\frac{\rho_E(s,s')}{\rho_\pi(s,s')}\right) \tag{29}$$

Proof. Note that the optimal discriminator is achieved when $D^*(s,s') = \frac{\rho_E(s,s')}{\rho_E(s,s')+\rho_\pi(s,s')}$ as used in [43] and shown in [44] Section 4 Proposition 1.

$$\begin{aligned}
f_\theta^*(s,s') &= \log(D^*(s,s')) - \log(1 - D^*(s,s')) && (30)\\
&= \log\left(\frac{\rho_E(s,s')}{\rho_E(s,s')+\rho_\pi(s,s')}\right) - \log\left(\frac{\rho_\pi(s,s')}{\rho_E(s,s')+\rho_\pi(s,s')}\right) && (31)\\
&= \log\left(\frac{\rho_E(s,s')}{\rho_\pi(s,s')}\right)
\end{aligned}$$

□

This shows that, by setting $r(s, s') = f_\theta(s, s')$, the recovered reward function is the log-ratio of state-transition occupancy measure from the expert to the policy.

Lemma B.2.2. *MPAIL minimizes a regularized KL divergence between the policy's state-transition occupancy measure and the expert's.*

Proof. Recall from Proposition B.1.1 that while solving for the RL objective, MPPI finds a policy of the form

$$\pi(a|s) \propto \overline{\pi}(a|s) e^{-\frac{1}{\beta}\overline{c}(s,a)} \tag{32}$$

Applying Proposition B.2.1, we plug $c(s, s') = -f^*_\theta(s, s')$ into Equation (9) to obtain

$$\pi^*(s, a) \propto \overline{\pi}(s, a) \exp\left(-\frac{1}{\beta}\sum_{s'\in\mathcal{S}} \mathcal{T}(s'|s)[\log \rho_\pi - \log \rho_E)]\right) \tag{33}$$

Note that when the policy distribution ρ_π matches the expert distribution ρ_E the exponential term collapses. Thus, when the occupancy measures match, the policy updates cease to have an effect and the optimization attains a fixed point.

In fact, we may note that for any fixed state $s \in \mathcal{S}$, the cost accumulated by the policy is

$$\sum_{s'\in\mathcal{S}} \rho_\pi(s, s')c^*(s, s') = \sum_{s'\in\mathcal{S}} \rho_\pi(s, s') \log\left(\frac{\rho_\pi(s, s')}{\rho_E(s, s')}\right) = \mathbb{KL}(\rho_\pi(s, \cdot)||\rho_E(s, \cdot)) \tag{34}$$

Finally, the MPPI objective can be written as

$$\min_{\pi\in\Pi} \mathbb{KL}(\rho_\pi||\rho_E) + \beta\, \mathbb{KL}(\pi\,||\,\overline{\pi}) \tag{35}$$

showing that the MPAIL procedure minimizes the entropy regularized KL Divergence between state-transition occupancy measures. In this sense, we have shown that MPAIL can be indeed classified as an AIL algorithm which seeks to match the expert's occupancy measure through an MPPI Policy.

$\square$

Remark B.2.3. *On Identifiability*. A question naturally arises about the limitations that being state-only imposes. If state transitions are deterministic and invertible, observing (s, s') is the same as observing the unique action a that caused it. Then $r(s, s') = r(s, a)$ and by the 1-1 correspondence of policies with state-action occupancy measures [11], the recovered policy becomes unique.

In general, this assumption has varying degrees of accuracy. When transitions are many to one or stochastic, multiple actions can produce the same transition (s, s'). Then $\rho_\pi(s, s') = \rho_\pi(s)\sum_{a\in\mathcal{A}} \pi(a|s)\mathcal{T}(s'|s, a)$ becomes a mixture over actions which induces a range of respective policies. For instance, if $\mathcal{T}(s'|s, a_1) = \mathcal{T}(s'|s, a_2)$ for actions a_1, a_2, the expert could perform either action and a state-transition based reward would not distinguish between them. Nonetheless, IRL is already an ill-posed problem due to the many to one relationships between policies, rewards, and demonstrations. Though the ambiguity is exacerbated by lack of demonstrated actions, it is still inherent to the problem.

C Implementation

In this section we provide further details about the algorithm implementation. Some features incorporated here are deemed well-known (i.e. spectral normalization) or not rigorously studied for statistical significance but included for completeness and transparency.

C.1 Regularization

Spectral Normalization. As often found in GAN and AIL surveys [32, 45], we corroborate that applying spectral normalization to the discriminator architecture appeared to have improved MPAIL training stability and performance. Application of spectral normalization to the value network did not appear to make a noticeable difference.

L2 Weight Regularization. Some experimentation was done with L2 weight regularization, but it was ultimately *not used for any simulation results*. Instead, usage of the weight regularization for the real experiment (Section 4.3) appeared to help stabilize training and allow for more reliable model selection and deployment.

C.2 Hyperparameters

Fundamentally derived from AIL and MPPI, we can etymologically partition hyperparameters into those induced by AIL (orange) and by MPPI (blue). Remaining non-highlighted parameters for this work are introduced and discussed below.

Hyperparameter	Value
Disc. optimizer (θ)	Adam ($\beta_1 = 0.5$, $\beta_2 = 0.999$)
Disc. learning rate	1e-4
Disc. hidden width	32
Disc. hidden layers	2
Disc. L2 coefficient	0 (sim), 0.001 (real)
Value optimizer (ϕ)	Adam ($\beta_1 = 0.9$, $\beta_2 = 0.999$)
Value learning rate	1e-3
Value hidden width	32
Value hidden layers	2
Value loss clip	0.2
Discount (γ)	0.99
Generalized return (λ)	0.95
Value max grad norm	1.0
Mini batches	3
Epochs	3
Trajectories (N)	512
Planning horizon (H)	10
Iterations (J)	5
Sampling variance (Σ)	$diag([0.3\ \ldots])$
Initial temperature (λ_0)	1.0
Markup/Discount (η)	1.01
Temp. decay rate	0.01
Minimum temp.	1e-5
Value:Disc update ratio	3:1 (sim), 1:1 (real)

Table 2: **MPAIL Hyperparameters**. Used across all experiments unless specified otherwise.

Temperature Decay. As noted in Section 3, we found that an initial temperature with a gradual decay (down to a minimum) was helpful in preventing early and unrecoverable collapse. The intuition for this decision is similar to that of decaying policy noise injection in many popular RL frameworks [46, 20], since the temperature is directly related to the variance of the optimized gaussian distribution. This component remains under investigation as its usage is not always necessary for meaningful convergence, but it is perhaps practically useful as it alleviates temperature tuning labor.

Value-to-Discriminator Update Ratio. Common to existing AIL (and GAN) implementations, MPAIL benefits from a balancing of generator and discriminator updates. Note that, like GANs, AIL tends to oscillate aggressively throughout training [47]. As MPAIL does not enforce a constrained policy update each epoch (as TRPO does [48]), the policy is exposed more directly to the discriminator's oscillations which can further hinder on-policy value estimation. A further converged value function is also theoretically more stationary from the perspective of π_{MPPI} as infinite-horizon MPPI.

Markup. A notable quirk discovered during implementation is the relationship between costs and rewards. While the two concepts are generally regarded as dual (with negation), it is worthwhile noting that discounting is not closed under negation. Meaning, it is not correct to apply the same discount factor to the costs as they are done to the summation of rewards in the return G_t. Consider the reward with a discount applied $r_1 = \gamma\, r(s, s')$ and the reward of a cost with a discount applied $r_2 = -\gamma\, c(s, s')$. Observe that for $\gamma < 1$, r_1 decreases while r_2 increases. Thus, when using costs, the H-step factor in the MPPI horizon, η, should not decrease over t'. In fact, when applying $\eta < 1$, we found that MPAIL does not meaningfully converge ever on the navigation task. Geldenbott and Leung [49] names the usage of $\eta > 1$ as a markup. In our case, we apply a similar empirical factor such that $\eta := 1/\gamma > 1$. While we suspect a more rigorous relationship between η and γ, we leave its derivation for future work. However, we remark that a reward-only variant of MPPI which precludes these relationships is equally possible as done in [50]. Costs are maintained in this work due to wider familiarity in practice [13, 16, 51, 52].

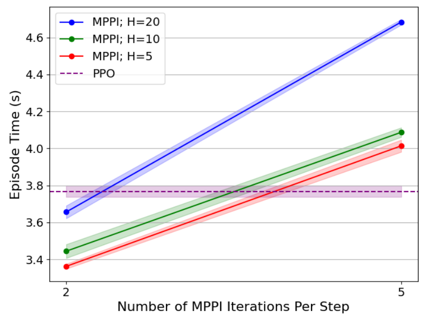

Figure 7: **Comparing "Inference" Times for Navigation Task.** Time taken to complete one episode of 100 timesteps with 64 parallel environments across varying horizon lengths and MPPI optimization iterations. PPO (in policy-based AIL) is used as implemented in the RSL library [53]. All training runs in this work are performed on an NVIDIA RTX 4090 GPU. Isaac Lab is chosen as our benchmarking and simulation environment due to its parallelization and robot learning extensions [54, 55].

```
MPAILPolicy initialized. Total number of params: 3779
Dynamics: 0
Sampling: 0
Cost: 3778
Temperature: 1
MPAILPolicy(
  (dynamics): KinematicBicycleModel()
  (costs): TDCost(
    (ss_cost): GAIfOCost(
      (reward): Sequential(
        (0): Linear(in_features=24, out_features=32, bias=True)
        (1): LeakyReLU(negative_slope=0.01)
        (2): Linear(in_features=32, out_features=32, bias=True)
        (3): LeakyReLU(negative_slope=0.01)
        (4): Linear(in_features=32, out_features=1, bias=True)
      )
    )
    (ts_cost): CostToGo(
      (value): Sequential(
        (0): Linear(in_features=12, out_features=32, bias=True)
        (1): ReLU()
        (2): Linear(in_features=32, out_features=32, bias=True)
        (3): ReLU()
        (4): Linear(in_features=32, out_features=1, bias=True)
      )
    )
  )
  (sampling): DeltaSampling()
)
```

Figure 8: **PyTorch [56] Model Architecture from Train Log.** MPAIL readily admits other well-studied components of the model-based planning framework (e.g. sampling, dynamics) [57, 58]. This work focuses on costing from demonstration.

C.3 Computation

MPAIL is crucially implemented to be parallelized across environments in addition to trajectory optimization. In other words, in a single environment step, each parallel environment independently performs parallelized sampling, rollouts, and costing entirely on GPU without CPU multithreading. MPPI also allows for customize-able computational budget, similar to [20] (see Figure 7). For the navigation and cartpole tasks, we find that online trajectory optimization implemented this way induces little impact on training times. In exchange, MPPI can be more space intensive due to model rollouts having space complexity of $\mathcal{O}(HN|\mathcal{S}|)$ per agent. On the navigation task benchmark settings, this is an additional 245 kB per agent or 15.7 MB in total for 64 environments. Figure 7 shows benchmarks on training times that demonstrate comparable times to PPO's policy inference. Overall, training runs for Section 4.4 between MPAIL and GAIL on the navigation task are comparable at about 45 minutes each for 500 iterations.

D Experimental Details

D.1 Real-Sim-Real Navigation

Setup Details. Before continuing with the discussion of our results, we provide further details about the setup of the experiment. The platform itself is an open-source MuSHR platform as detailed in [38]. Notably, the compute has been replaced with an NVIDIA Jetson Orin NX as mentioned in Section 4.3. Poses (position, orientation; $[x \quad y \quad z \quad r \quad p \quad y]$) are provided by a motion-capture system at a rate of 20 Hz. Velocities are body-centric as estimated by onboard wheel encoders ($\mathbf{v} = v_x\mathbf{b}_1 + v_y\mathbf{b}_2 + v_z\mathbf{b}_3$, such that $\mathbf{b}_1$ points forward, $\mathbf{b}_2$ points left, and $\mathbf{b}_3$ completes the right-hand frame; basis vectors are rigidly attached to the vehicle [13]). Note that the vehicle is operated without slipping nor reversing such that $v_y \approx v_z \approx 0$ and $v_x > 0$ [33]. The recorded states used for the expert demonstration data is 240 timesteps long. Altogether, the data can be written as $s_E \in \{(x_t, y_t, z_t, v_{x,t}, v_{y,t}, v_{z,t})\}_{t=1}^{240}$.

A remark: GAIL for this task is necessarily implemented with "asymmetry" between actor and reward. Since, the discriminator must receive as input expert observations s_E while the agent is provided (r, p, y) in addition to observations in s_E. In theory, there should be no conflict with the IRLfO (Equation (1)) formulation as this remains a valid reward but on a subset of the state.

Additional Discussion of Results.

Note that the direction of travel cannot be uniquely determined by a single state s due to the partially observable body-centric *velocity.* Rather, only with the state-transition (s_E, s'_E) is it possible to deduce the direction of travel. For instance, consider a simplified hand-designed cost using the partially observable expert data $c(s, s'|s_E, s'_E)$. A reference vector can be computed through the difference of positions between s' and s then scaled by the demonstrated velocities: $c(s, s'|s_E, s'_E) := \|^{\mathcal{I}}\mathbf{v}(s, s') - v_{xE}[(x'_E - x_E)\mathbf{e}_1 + (y'_E - y_E)\mathbf{e}_2]\|_2$ where $\mathbf{e}_i$ are global basis vectors for global frame $\mathcal{I}$ and $^{\mathcal{I}}\mathbf{v}(s, s')$ is the robot velocity in $\mathcal{I}$. Of course, this example assumes the ability to correctly choose the corresponding (s_E, s'_E) pair for input (s, s') out of the entirety of the expert dataset d^E. It should be clear that partial observability and state-transitions play critical roles in the recovery of this non-trivial cost function. This experiment presents a necessary challenge towards practical AIL [32] and scalable Learning-from-Observation (LfO) [27].

IRL-MPC was evaluated across three ablations: (a) reward-only, (b) value-only, and (c) reward-and-value. The results in Figure 5 reflect the performance of (a) reward-only. The other implementations were distinctly worse than (a) and frequently devolved into turning in circles much like GAIL.

In both cases of GAIL and MPAIL, we find that the agents occasionally travel counter-clockwise (where the expert travels clockwise) during training, suggesting that (s_E, s'_E) appears close to (s'_E, s_E) through the discriminator. As the data is collected through real hardware, it is suspected that state estimation noise introduces blurring between states that are separated by only 50 ms. GAIL is otherwise known to perform poorly in the existence of multi-modal data [59]. This is further corroborated by its unstable performance on the navigation benchmark. And, to the best of our knowledge, similar Real-Sim-Real applications of AIL appear sparse if existent at all. Adjacent works which use real demonstration data but train in real include [37, 19]. Even while training in real, GAIL's performance drops signficantly ($90\% \to 20\%$) when presented with imperfect demonstrations for even straightforward tasks like reaching [37, 19]. These observations might suggest why the GAIL discriminator is unable to learn meaningfully in simulation and produces a poor policy.

Meanwhile, MPAIL's success and IRL-MPC's improvement over GAIL is attributed to model-based planning capabilities. If the robot should find itself away from the expert distribution, the online planner enables the agent to sample back onto the demonstration. This becomes especially important when the demonstration data is severely under-defined as in this partially observable setting, which results in positive reward signal in most states in the environment. This is further supported by the fact that discriminator predictions are comparatively uncertain when training over real data. Recall that low discriminator confidence is reflected by low magnitude logit f_θ (cost) predictions. For reference, real experiment cost values are in the range of $(-0.022, -0.0180)$ whereas cost values in benchmarking runs with synthetic demonstrations are in the range of $(-3, 3)$. On real data, the discriminator also required more frequent updates to provide more reliable signals (see Table 2).

D.2 Simulated Navigation Task Details

Reward and Data. The exact form of the reward used for training PPO and for metrics is given by

$$r(s) := \sqrt{10^2 + 10^2} - \sqrt{(x - 10)^2 + (y - 10)^2}.$$

Figure 9 visualizes four demonstrations from the converged PPO "expert" policy. Additional demonstrations are generated by playing more environments from this policy for one episode such that each demonstration is distinct. Each episode is 100 timesteps long, where each timestep is 0.1 seconds.

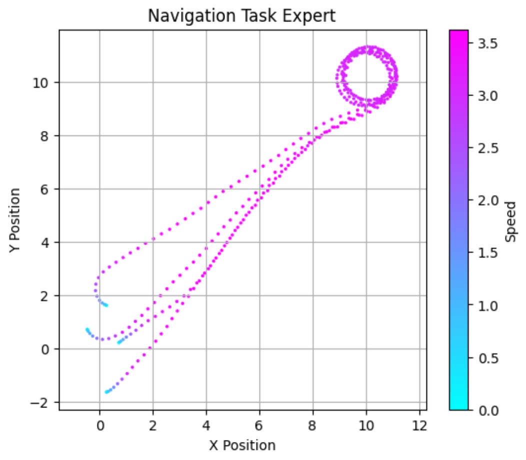

Figure 9: **Visualization of Four Expert Trajectories in Navigation Task.** Cars are initialized around $(0, 0)$ and navigate towards $(10, 10)$ where the "circling" behavior begins, as discussed in Section 4.1.

OOD Experiment. OOD Energy in Figure 4 is computed as described by Liu et al. [35]. Namely, with respect to the expert data d^E, we fit a reference distribution using $\tilde{P}_E = \mathcal{N}(\bar{\mu}_E, \bar{\Sigma}_E)$. Then, the OOD energy is given by $E(s; p_E) = \log p_E(s)$. Some limitations of this procedure can be observed given that ID points for GAIL do not receive as much reward as one might expect. However, this remains reasonable considering that the GAIL policy may forget ID behavior, which can be seen in Figure 3 by agents clearly ID remaining static throughout the episode. Future work might better explore quantifying OOD towards measuring AIL generalization through direct usage of the discriminator.

D.3 Predictive Model Learning Towards Generalizable MPAIL

For tasks beyond navigation (see also Section D.5 for the Ant environment), planning rollouts were generated from a deterministic dynamics model $f_\psi(s, a)$ learned entirely online. The dynamics model was trained to minimize the mean squared error between the predicted and observed s_{t+1}, given s_t and a_t. The loss being optimized can be written as:

$$\hat{s}_{i+1} = f_\psi(s_i, a_i), \quad L = \frac{1}{H_B} \sum_{s,a \in B} (s_{i+1} - \hat{s}_{i+1})^T (s_{i+1} - \hat{s}_{i+1}) \tag{36}$$

with model parameters ψ, transition buffer B, and a mini-batch of size H_B sampled from B. If used, the update for the model occurs after line 4 in Algorithm 1.

Dynamics Model Hyperparameter	**Value**
Optimizer	Adam($\beta_1 = 0.9, \beta_2 = 0.999$)
Learning rate	1e-3
LR decay rate	0.9
LR decay frequency (ep.)	25 (Ant), 15 (Cartpole)
Min. LR	1e-6
Hidden width	256 (Ant), 64 (Cartpole)
Hidden layers	3

Table 3: **Dynamics Learning Hyperparameters.**

Training augmentations. Several training augmentations were made to improve model accuracy and stability. A transition replay buffer, which stored transitions from multiple episodes, was used to train the dynamics model for multiple epochs during each MPAIL training iteration. After each episode, the buffer was updated by randomly replacing old transitions with those from the latest episode. This off-policy buffer helped stabilize training and prevent overfitting when training using multiple epochs. Furthermore, applying a step-based learning rate decay improved convergence speed. Dynamics model-specific hyperparameters are listed in Table 3.

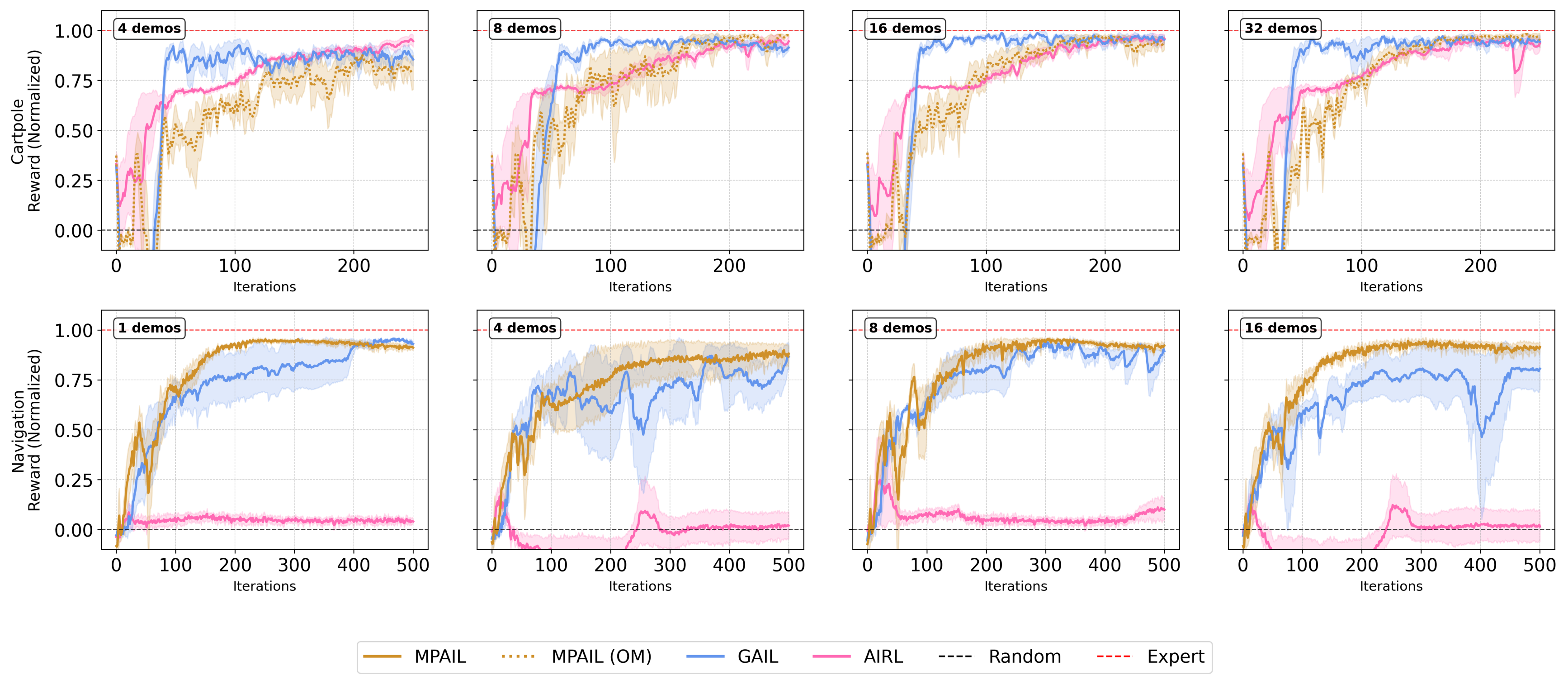

Figure 10: Benchmark results from Figure 6 de-aggregated across number of demonstrations. MPAIL results for the Cartpole task is differentiated **MPAIL (OM)** to clarify that its **M**odel is learned fully **O**nline.

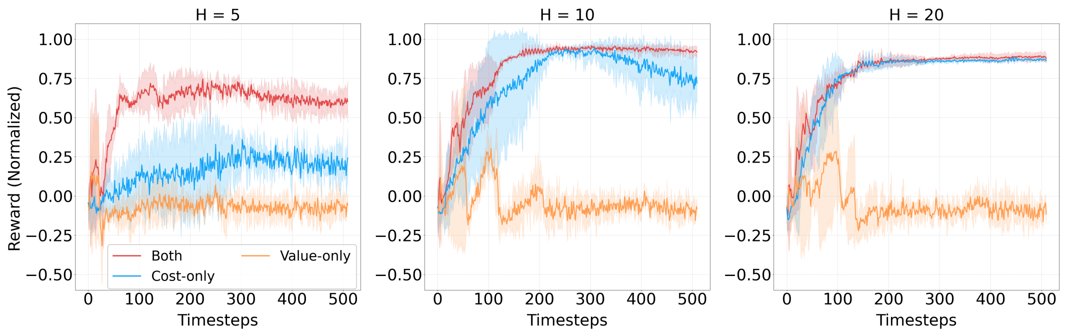

Figure 11: **Ablating single-step costs c_θ and value V_ϕ across different Horizon (H) lengths.** "Cost-only" experiments are performed by not evaluating V_ϕ on the final state in each model rollout. "Value-only" experiments are performed by not evaluating c_θ on the H-step state-transitions. See Algorithm 2, line 12 for exact usage.

D.4 Ablations

Figure 11 shows the results of an ablation study, investigating the effect of including costs or values in the MPAIL formulation. We find that including both is necessary for reasonable behavior across varying horizon lengths. We observe that value-only planning can quickly improve but is highly unstable and is unreliable as a generator. As expected, cost-only planning performs progressively better at longer planning horizons. However, should the agent find itself off-distribution, it is not able to return to the distribution until it randomly samples back in, which may potentially never occur. For instance, many agents which are initialized facing the opposite direction drive randomly without ever returning to the distribution. Without a value function guiding the agent, the discriminator (i.e. cost) does not provide a significant reward signal for returning to the distribution. This can be observed in the $H = 10$ plot where the performance of cost-only planning quickly drops as the discriminator is further refined on the expert data, decreasing the likelihood of randomly sampling into distribution. In this sense, the combination of cost and value operates as intended: *costing* is necessary for defining and staying inside the expert distribution, while *value* is necessary for generalizing the reward beyond the support of the expert elsewhere in the environment.

D.5 Towards High-Dimensional Tasks from Demonstration with Sampled-Based MPC

Figure 12 provides an experiment of MPAIL training an agent in the Isaac Lab implementation of the Ant-v2 environment as a step towards high-dimensional applications. As expected, MPPI's (vanilla) sampling procedure struggles to be competitive with policy-based optimization in higher-dimensional spaces. However, MPAIL demonstrates signs of life in enabling MPPI to optimize a state space otherwise considered extremely challenging for sample-based planning. Note that Isaac Lab's Ant implementation prescribes a 60-dimensional observation and 8-dimensional action space, rather than Mujoco's 26-dimensional observation. Thus, the space is 120-dimensional for costing $c_\theta(s, s')$ and 80-dimensional for MPPI with a planning horizon of 10 timesteps. Despite this, a learned cost is capable of guiding a real-time vanilla MPPI optimization to execute walking behaviors in the ant task, albeit slower, even from few demonstrations.

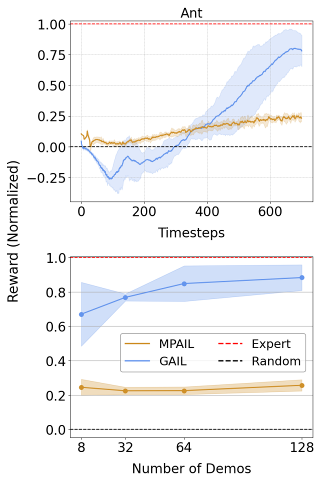

Figure 12: **Isaac Lab Ant-v2 Experiment**.

"Remembering" locally optimal policies through a learned policy-like proposal distribution may help planning capabilities generalize to higher-dimensional spaces. Additionally, modeling dynamics in latent-space and using model ensembles have been shown to significantly improve performance in model-based reinforcement learning [50] and are promising directions for future work for high-dimensional tasks. Finally, Figure 8 illustrates a key takeaway of this framework: learning costs through MPAIL remains orthogonal to other works which seek to improve sample-based MPC through sampling [41, 58, 60], optimization [58, 61], and dynamics [20]. We believe that integration of developments in MPC along with application-specific cost regularization [52] may be critical for exploring the full potential of planning from observation.