In *Proceedings of The IEEE/RSJ International Conference on Intelligent Robots and Systems* (IROS 2021), Prague, Czech Republic, September 2021

DEALIO: Data-Efficient Adversarial Learning for Imitation from Observation

Faraz Torabi[1], Garrett Warnell[2] and Peter Stone[3]

***Abstract*— In imitation learning from observation (IfO), a learning agent seeks to imitate a demonstrating agent using only *observations* of the demonstrated behavior without access to the control signals generated by the demonstrator. Recent methods based on adversarial imitation learning have led to state-of-the-art performance on IfO problems, but they typically suffer from high sample complexity due to a reliance on data-inefficient, model-free reinforcement learning algorithms. This issue makes them impractical to deploy in real-world settings, where gathering samples can incur high costs in terms of time, energy, and risk. In this work, we hypothesize that we can incorporate ideas from model-based reinforcement learning with adversarial methods for IfO in order to increase the data efficiency of these methods without sacrificing performance. Specifically, we consider time-varying linear Gaussian policies, and propose a method that integrates the linear-quadratic regulator with path integral policy improvement into an existing adversarial IfO framework. The result is a more data-efficient IfO algorithm with better performance, which we show empirically in four simulation domains: using far fewer interactions with the environment, the proposed method exhibits similar or better performance than the existing technique.**

I. Introduction

Reinforcement learning (RL) [1], [2], [3] is a machine learning paradigm that makes it possible for artificial, autonomous agents to learn tasks using their own experience. Broadly speaking, RL agents repeatedly observe the state of the environment they inhabit, perform an action within that environment, and receive feedback based on the utility of taking that action with respect to a particular task that they are trying to perform. Agents learn how to successfully perform tasks by modifying their action-selection policy so as to induce behavior that optimizes the expected cumulative feedback they receive. While RL methods have led to artificial agents successfully learning to perform all kinds of tasks, the problem of designing good feedback, or cost, functions is still one that must be solved by humans, and doing so is difficult since it typically requires a great deal of domain knowledge.

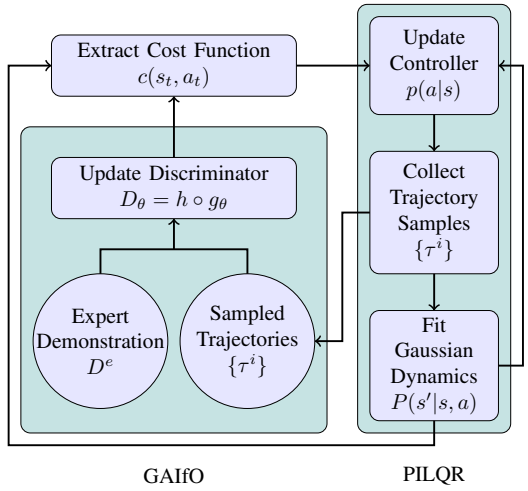


Fig. 1. A block diagram representing the learning flow of DEALIO. A time-varying Gaussian controller $p(a|s)$ is initialized and used to collect trajectories $\{\tau^i\}$ which are then used to fit a linear Gaussian dynamics model $P(s'|s,a)$. The collected data $\{\tau^i\}$ is also used with the demonstration data D^e to train a discriminator D_θ which is a composition function of a quadratic function $h(., s_t, s_{t+1})$ and a neural network $g_\theta(s_t, s_{t+1})$. The discriminator D_θ and the dynamics model $P(s'|s,a)$ are then used to extract the cost function $c(s,a)$ which is then used to update the controller $p(a|s)$.

The paradigm of imitation learning (IL) [4], [2], [5], on the other hand, sidesteps the cost function design problem by instead using *demonstrations* of the desired behavior in order to guide the machine learning process. Demonstrating tasks is often much easier for humans compared to designing a function that can provide detailed feedback to the agent in arbitrary situations. In traditional IL, however, the demonstration data must include information about both the states and actions experienced by the demonstrator. The requirement of actions in particular prevents the agent from being able to learn from cheaper, passive resources such as YouTube videos, which typically contain only state information in the form of *observations* of the demonstrator.

To remove the need for action information, the research community has recently given a great deal of attention to the related paradigm of imitation learning from observation (IfO) [6], [7], which considers explicitly the case where artificial agents seek to learn behaviors from demonstrations consisting only of state information. One class of algorithms that has achieved state-of-the-art performance on IfO problems relies on techniques from adversarial learning [8], [9], [10]. For

*This work has taken place in the Learning Agents Research Group (LARG) at UT Austin. LARG research is supported in part by NSF (CPS-1739964, IIS-1724157, NRI-1925082), ONR (N00014-18-2243), FLI (RFP2-000), ARO (W911NF-19-2-0333), DARPA, Lockheed Martin, GM, and Bosch. Peter Stone serves as the Executive Director of Sony AI America and receives financial compensation for this work. The terms of this arrangement have been reviewed and approved by the University of Texas at Austin in accordance with its policy on objectivity in research.

[1]First Author is with the Department of Computer Science, The University of Texas at Austin, USA faraztrb@cs.utexas.edu

[2]Second Author is with Army Research Laboratory, USA and the Department of Computer Science, The University of Texas at Austin, USA garrett.a.warnell.civ@mail.mil

[2]Third Author is with the Department of Computer Science, The University of Texas at Austin and Sony AI, USA pstone@cs.utexas.edu

example, generative adversarial imitation from observation (GAIfO) [9], [11] uses adversarial learning to bring the state-transition distribution of the imitator closer to that of the demonstrator. As with other adversarial IfO approaches, GAIfO relies on a model-free RL algorithm as part of the learning process, in which the goal is to learn a policy directly from samples without forming or leveraging any intermediate representations of the environment or cost function. These model-free techniques typically exhibit extremely high sample complexity, which can prove problematic in real-world settings in which each sample required for learning incurs a high cost in terms of time, energy, and/or risk.

One classical way in which the RL community has dealt with poor sample efficiency is through the use of model-based RL. Unlike model-free techniques, model-based RL algorithms make assumptions about the environment dynamics such as it being linear or differentiable. and learn a model of the dynamics during the learning process. Model-free RL algorithms are known to perform well at the cost of data inefficiency [12], [13], i.e., the final learned policy is able to perform close to optimality, but learning it requires many interactions with the environment. On the other hand, model-based algorithms are known to be sample-efficient, but, since they rely on assumptions to build models, they typically result in sub-optimal policies, especially in environments that exhibit complex dynamics [14]. Some RL algorithms have been proposed to take advantage of the benefits of both model-based and model-free algorithms. One approach of this kind is PILQR [15] which has been shown both to achieve impressive task performance and to be sample-efficient enough to be deployed on physical robots.

In this paper, we propose to address sample inefficiency in adversarial IfO algorithms by integrating ideas from model-based RL. In particular, we propose to integrate the sample-efficient RL updates from PILQR with the high-performing GAIfO algorithm for IfO. The resulting algorithm, Data-Efficient Adversarial Learning for Imitation from Observation (DEALIO), is able to learn a time-varying linear Gaussian controller that can imitate a demonstrator using state-only demonstrations of a behavior. We evaluate DEALIO in several simulation domains and compare it with GAIfO. Our results show that DEALIO can achieve good imitation performance using far fewer environment interactions during learning. Moreover, in some cases, DEALIO is even able to outperform GAIfO.[1]

II. Related Work

We now review related work in the areas of sample-efficient reinforcement learning and imitation learning from observation.

A. Reinforcement Learning

Broadly speaking, reinforcement learning algorithms can be categorized as model-based or model-free. Model-free algorithms are known for their ability to handle complex dynamics [12], [13], while model-based algorithm are known for their data efficiency [14], [17], [15]. With the goal of trying to reap the advantages of both techniques, many algorithms have been proposed that seek to combine the advantages of each [18], [19]. The particular work that DEALIO is most related to is PILQR which combines path integral policy improvement (PI^2) [20] and the linear-quadratic regulator (LQR) [17].

DEALIO is similar to the work mentioned above in that it is intended to learn high quality policies with small number of interactions with the environment. However, it is different in that the methods mentioned above are reinforcement learning algorithms and require a cost function; while DEALIO is an IfO algorithm that imitates from a demonstrator.

B. Imitation Learning from Observation (IfO)

IfO is a task faced by machine learners that seek to imitate state-only behavior demonstrations [7]. This topic has received a great deal of attention over the years, and several different types of IfO algorithms have been developed [21], [6], [22], [23]. The work presented here is most related to adversarial imitation from observation algorithms. These algorithms typically attempt to find imitation policies that induce behaviors that result distribution matching with the demonstrator with respect to either *states* [8], [24] or *state transitions* [9], [11]. One algorithm in particular that our work builds off of is Generative Adversarial Imitation from Observation (GAIfO) [9], [11], which exhibits good final performance on benchmark tasks, but also suffers from poor sample complexity. Other IfO work includes that which explores the choice of imitator state representation [25] and work that has explored cases when the imitator and demonstrator exhibit different action spaces [26], viewpoints [27], or embodiment mismatch [28].

The algorithm we propose here, DEALIO, is similar to the ones mentioned above in that our algorithm is also an adversarial imitation from observation algorithm. However, DEALIO seeks to resolve high sample complexity issues by leveraging data-efficient reinforcement learning algorithms. In this respect, our work is similar to very recent work on off-policy IfO [29], though we pursue a fundamentally-different approach by considering on-policy techniques. Additionally, while multiple approaches have recently sought to address the issue of high sample complexity in adversarial imitation learning [30], [31], these approaches have been developed in the setting of demonstrations that consist of states and actions. In contrast, our work here focuses on situations in which demonstrations consists of state information only.

III. Background

The proposed method is developed in the specific context of several techniques within the existing literature, which we review here in detail.

A. Reinforcement Learning

Reinforcement learning problems are posed in the context of a Markov Decision Process (MDP), i.e., a tuple $\mathcal{M} =$

[1] An earlier version of this work was presented in ICML Workshop on Imitation, Intent, and Interaction (I3) [16]. This paper however, is drastically different both in the design of the proposed algorithm and the experiments.

$\{\mathcal{S}, \mathcal{A}, P, c\}$, where $\mathcal{S}$ and $\mathcal{A}$ are state and action spaces, respectively, P is a transition probability distribution, and c is the cost function. At a given time instant, a decision-making agent operating within M finds itself in state $s \in \mathcal{S}$ and takes an action $a \in \mathcal{A}$ based on a policy $\pi : \mathcal{S} \rightarrow \mathcal{A}$. As a result, agent moves to a new state $s' \in \mathcal{S}$ with probability $P(s'|s, a)$, at which point the agent also receives feedback $c(s, a)$ based on the cost function $c : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$. The goal of reinforcement learning agents is to use their own experience to learn a policy that results in a behavior that incurs minimal expected cumulative cost.

1) The Linear-Quadratic Regulator (LQR): The Linear-Quadratic Regulator (LQR) [32], [33] is a sample-efficient, model-based reinforcement learning algorithm that makes several strict assumptions regarding the features of the system such as the environment transition distribution and the cost function. Specifically, LQR assumes that the environment dynamics are linear, i.e.,

$$s_{t+1} = f(s_t, a_t) = F_t \begin{bmatrix} s_t \\ a_t \end{bmatrix} + f_t \ , \tag{1}$$

where F_t and f_t are known matrices and vectors, respectively. LQR also assumes the cost function is quadratic, i.e.,

$$c(s_t, a_t) = \frac{1}{2} \begin{bmatrix} s_t \\ a_t \end{bmatrix}^T C_t \begin{bmatrix} s_t \\ a_t \end{bmatrix} + \begin{bmatrix} s_t \\ a_t \end{bmatrix}^T c_t \ , \tag{2}$$

where C_t and c_t are also known matrices and vectors, respectively. Under these assumptions, LQR seeks an optimal controller (policy) of the form

$$a_t = K_t s_t + k_t \ , \tag{3}$$

where K_t and k_t are functions of the model parameters F_t, C_t, f_t, and c_t that can be computed for each time step using a procedure known as backward recursion.

LQR can also be used under the assumption of linear Gaussian dynamics, i.e.,

$$\begin{aligned} s_{t+1} \sim P(s'|s, a) &= \mathcal{N}(f(s_t, a_t), \Sigma_t) \\ &= \mathcal{N}(F_t \begin{bmatrix} s_t \\ a_t \end{bmatrix} + f_t, \Sigma_t) \ . \end{aligned} \tag{4}$$

In this case, the solution is still a controller of the form specified in Equation 3, where K_t and k_t can again be calculated using LQR backward recursion.

Note that, to compute the controllers as described above, one must know the dynamics model parameters F_t and f_t. If the dynamics model is unknown, methods have been proposed in which it can be learned through experience [34]. However, for complex dynamics, one global model is generally not sufficient for the entire state space, and so local dynamics models are typically learned instead, i.e., new linear Gaussian dynamics models are fit to the available data for each time step.

2) Combining Path Integral Policy Improvement and LQR (PILQR): In trying to derive the benefits of both model-free and model-based RL techniques, the PILQR algorithm [15] integrates model-based updates from LQR with fitted linear models and Path Integral Policy Improvement (PI^2) [20], which is a model-free RL algorithm based on stochastic optimal control. PILQR removes the quadratic cost requirement of LQR, instead requiring only that the cost function be twice differentiable. PILQR forms a quadratic approximation of the cost, and makes policy updates by first using the iterative Linear-Quadratic Regulator (iLQR) approach [32] with the quadratic approximation, and then performing a subsequent policy update using PI^2 on the residual cost.

B. Imitation Learning

Imitation learning (IL) is a machine learning paradigm in which autonomous agents seek to learn behaviors without having access to explicit cost feedback as in RL, instead operating in the context of a MDP without cost, $\mathcal{M} \backslash c$. In this setting, the agent instead has access to some demonstrations of desirable behavior. These demonstrations consist of the states and actions of the demonstrator, $D^e = \{\tau_i^e\}$, where $\tau_i^e = \{(s^e, a^e)\}_i$. Requiring access to the actions of the demonstrator however, makes it impossible for the agents to learn from the cases where videos of the demonstrator are the only available resources.

C. Imitation Learning from Observation (IfO)

Imitation learning from observation (IfO) removes the requirement in IL that the demonstrations have to contain action information. That is, in IfO, demonstrations consist of state information only, i.e., $D^e = \{\tau_i^e\}$, where $\tau_i^e = \{(s^e)\}_i$.

Our work builds off of an IfO algorithm called Generative Adversarial Imitation from Observation (GAIfO) [9], [11], [25] which is inspired by Generative Adversarial Networks (GANs) [35]. GAIfO attempts to learn a policy such that the imitator's state-transition distribution matches that of the demonstrator. The algorithm works as follows. First, a random neural network policy π_ϕ is initialized as the imitator's policy, which is used by the imitator to generate state trajectories $\{\tau_i^i\}$ where $\tau_i^i = \{(s^i)\}_i$. Next, a neural network discriminator is trained to discriminate between the state transitions provided by the demonstrator and the state transitions generated by the imitator. Specifically, the discriminator training is done using supervised learning with the following loss function:

$$\begin{aligned} &- \Big(\mathbb{E}_{\{\tau_i^i\}}[\log(D_\theta(s_t, s_{t+1}))] \\ &+ \mathbb{E}_{\{\tau_i^e\}}[\log(1 - D_\theta(s_t, s_{t+1}))] \Big) \ , \end{aligned} \tag{5}$$

where D is the discriminator network parameterized by θ and s_t and s_{t+1} are two consecutive states. Finally, the model-free RL algorithm Trust Region Policy Optimization (TRPO) [13] is used to update the imitator's policy using the cost function

$$\mathbb{E}_{\{\tau_i^i\}}[\log(D_\theta(s_t, s_{t+1}))] \ . \tag{6}$$

While GAIfO has shown very promising performance, it also exhibits high sample complexity.

IV. DEALIO

We now introduce DEALIO, an algorithm for IfO that has the explicit goal of making adversarial imitation learning from observation techniques more sample efficient through the use of model-based RL. DEALIO is based on the GAIfO algorithm discussed in Section III-C, in which a neural network discriminator attempts to distinguish between state transitions generated by the current imitation policy and state transitions generated by the demonstrator. The discriminator's output is used as the cost function that drives imitation policy learning using the model-free RL algorithm TRPO. To improve sample efficiency, DEALIO aims to replace TRPO with the PILQR algorithm that computes policy updates using a combination of model-based and model-free RL.

Making this change to GAIfO is nontrivial due to the functional form of the cost function required by PILQR. First, PILQR requires a quadratic approximation of the cost function in order to compute updates, whereas the cost function used in GAIfO is specified by the discriminator, which is a continually updating deep neural network. As mentioned in Section III-A.2, it is not required for the cost function to be in the form of Equation 2, instead it only needs to be twice-differentiable. Therefore, we aim to develop a cost function in the form of

$$c(s_t, a_t) = \frac{1}{2} \begin{bmatrix} s_t \\ a_t \end{bmatrix}^T C_t \begin{bmatrix} s_t \\ a_t \end{bmatrix} + \begin{bmatrix} s_t \\ a_t \end{bmatrix}^T c_t + cc_t \ . \quad (7)$$

for which the quadratic approximation would be the first two terms of the right hand side.[2] Second, PILQR requires a cost function over both states and actions (as suggested by Equation 7), whereas, because GAIfO is an IfO technique, the discriminator that specifies the cost function is a function of state information only.

Overcoming these challenges and learning a cost function presented in Equation 7 requires us to find a method by which we can compute C_t, c_t, and cc_t in the context of the GAIfO algorithmic structure. To do so, we first propose to modify the structure of the discriminator by considering it to be a composition of two functions, $g_\theta(s_t, s_{t+1})$ and $h(., s_t, s_{t+1})$. The first function, $g_\theta(s_t, s_{t+1})$, is a neural network (with parameters θ) that takes state transitions as inputs and outputs the following quantities:

- The elements of a matrix $C^{ss}(s_t, s_{t+1})$
- The elements of a vector $c^{ss}(s_t, s_{t+1})$
- A constant $cc^{ss}(s_t, s_{t+1})$

The second function, $h(., s_t, s_{t+1})$, is a quadratic function in which the outputs of $g_\theta(s_t, s_{t+1})$, i.e. $C^{ss}(s_t, s_{t+1})$ and $c^{ss}(s_t, s_{t+1})$, can be used as its matrix and vector and the overall function composition becomes

$$\begin{aligned} D_\theta(s_t, s_{t+1}) &= h(g_\theta(s_t, s_{t+1}), s_t, s_{t+1}) \quad (8) \\ &= \frac{1}{2} \begin{bmatrix} s_t \\ s_{t+1} \end{bmatrix}^T C^{ss}(s_t, s_{t+1}) \begin{bmatrix} s_t \\ s_{t+1} \end{bmatrix} \\ &+ \begin{bmatrix} s_t \\ s_{t+1} \end{bmatrix}^T c^{ss}(s_t, s_{t+1}) \ . \end{aligned}$$

which yield the discriminator we use in DEALIO. By imposing this structure on the discriminator, we now have a quadratic cost function, $h(g_\theta(s_t, s_{t+1}), s_t, s_{t+1})$, as required by PILQR. However, two issues still remain. First, $h(g_\theta(s_t, s_{t+1}), s_t, s_{t+1})$ is still a function of state transitions rather than state-action pairs. Second, $C^{ss}(s_t, s_{t+1})$ and $c^{ss}(s_t, s_{t+1})$ are functions of states rather than time.

In order to find a quadratic cost function with state-action pairs as input, we use the linear dynamics model assumed by PILQR in Equation 1, which allows us to rewrite s_{t+1} in $h(g_\theta(s_t, s_{t+1}), s_t, s_{t+1})$ as a function of s_t and a_t. To do so, we first partition quantities from Equations 1 and 8 as follows:

$$F_t = \begin{bmatrix} F_{s_t} \\ F_{a_t} \end{bmatrix} \ , \quad (9)$$

$$C^{ss}(s_t, s_{t+1}) = \begin{bmatrix} C^{ss}_{s_t, s_t} & C^{ss}_{s_t, s_{t+1}} \\ C^{ss}_{s_{t+1}, s_t} & C^{ss}_{s_{t+1}, s_{t+1}} \end{bmatrix} \ , \quad (10)$$

and

$$c^{ss}(s_t, s_{t+1}) = \begin{bmatrix} c^{ss}_{s_t} \\ c^{ss}_{s_{t+1}} \end{bmatrix} \ . \quad (11)$$

Next, substituting s_{t+1} from Equation 1 into Equation 8 and doing some linear algebra, we can write

$$\begin{aligned} h(s_t, a_t) &= \frac{1}{2} \begin{bmatrix} s_t \\ a_t \end{bmatrix}^T C^{sa}(s_t, s_{t+1}) \begin{bmatrix} s_t \\ a_t \end{bmatrix} \quad (12) \\ &+ \begin{bmatrix} s_t \\ a_t \end{bmatrix}^T c^{sa}(s_t, s_{t+1}) \ , \end{aligned}$$

where the new matrix and vector, $C^{sa}(s_t, s_{t+1})$ and $c^{sa}(s_t, s_{t+1})$, respectively, are given by

$$C^{sa}(s_t, s_{t+1}) = \begin{bmatrix} C^{sa}_{s_t, a_t} & C^{sa}_{s_t, a_t} \\ C^{sa}_{a_t, s_t} & C^{sa}_{a_t, a_t} \end{bmatrix} \ , \quad (13)$$

and

$$c^{sa}(s_t, s_{t+1}) = \begin{bmatrix} c^{sa}_{s_t} \\ c^{sa}_{a_t} \end{bmatrix} \ , \quad (14)$$

and the individual partition terms above are computed as

$$\begin{aligned} C^{sa}_{s_t, s_t} &= C^{ss}_{s_t, s_t} + F^T_{s_t} C^{ss}_{s_{t+1}, s_t} + C^{ss}_{s_t, s_{t+1}} F_{s_t} \quad (15) \\ &+ F^T_{s_t} C^{ss}_{s_{t+1}, s_{t+1}} F_{s_t} \ , \end{aligned}$$

$$C^{sa}_{s_t, a_t} = C^{ss}_{s_t, s_{t+1}} F_{a_t} + F^T_{s_t} C^{ss}_{s_{t+1}, s_{t+1}} F_{a_t} \ , \quad (16)$$

$$C^{sa}_{a_t, s_t} = F^T_{a_t} C^{ss}_{s_{t+1}, s_t} + F^T_{a_t} C^{ss}_{s_{t+1}, s_{t+1}} F_{s_t} \ , \quad (17)$$

[2] Another option is to use a neural network discriminator (the same as GAIfO) and take the second order Taylor expansion of the neural network to calculate the quadratic approximate of the cost function. This approach also meets the requirements of PILQR. We implemented this approach and saw that DEALIO outperforms it by a large margin. We posit that the reason is that the manifold learned using the neural network is very complex that its second order Taylor expansion is not meaningful enough.

Algorithm 1 DEALIO

1: Initialize controller $p(a|s)$
2: Initialize a neural network discriminator D_θ with random parameter θ
3: Collect demonstration trajectories $D^e = \{\tau^e\}$
4: **while** Controller Improves **do**
5: Execute the controller to collect state-action trajectories $\{\tau_i^i\}$
6: Fit a linear Gaussian dynamics model $P(s'|s,a)$
7: Update D_θ using loss

$$-\Big(\mathbb{E}_{\{\tau_i^i\}}[\log(D_\theta(s_t, s_{t+1}))] + \mathbb{E}_{\{\tau_i^e\}}[\log(1 - D_\theta(s_t, s_{t+1}))]\Big)$$

and store $C^{ss}(s_t, s_{t+1})$, $c^{ss}(s_t, s_{t+1})$, and $cc^{ss}(s_t, s_{t+1})$
8: Compute the cost $c(s_t, a_t)$ by calculating C_t, c_t, and cc_t
9: Perform a PILQR update to update the controller $p(a|s)$

$$C^{sa}_{a_t,a_t} = F^T_{a_t} C^{ss}_{s_{t+1},s_{t+1}} F_{a_t} \ , \tag{18}$$

$$c^{sa}_{s_t} = \frac{1}{2} C^{ssT}_{s_{t+1},s_t} f_t + \frac{1}{2} C^{ss}_{s_t,s_{t+1}} f_t + \frac{1}{2} F^T_{s_t} C^{ssT}_{s_{t+1},s_{t+1}} f_t + \frac{1}{2} F^T_{s_t} C^{ss}_{s_{t+1},s_{t+1}} f_t + c^{ss}_{s_t} + F^T_{s_t} c^{ss}_{s_{t+1}} \ , \tag{19}$$

and

$$c^{sa}_{a_t} = \frac{1}{2} F^T_{a_t} C^{ssT}_{s_{t+1},s_{t+1}} f_t + \frac{1}{2} F^T_{a_t} C^{ss}_{s_{t+1},s_{t+1}} f_t + F^T_{a_t} c^{ss}_{s_{t+1}} \ . \tag{20}$$

Finally, in order to find cost function parameters C_t, c_t, and cc_t (as shown in Equation 7) to be functions of time only (i.e., independent of states), we use the mean state transition at time t as input to $C^{sa}(\cdot,\cdot)$, $c^{sa}(\cdot,\cdot)$, and $cc^{ss}(\cdot,\cdot)$, i.e.,

$$C_t = C^{sa}(\bar{s}_t, \bar{s}_{t+1}) \ , \tag{21}$$

$$c_t = c^{sa}(\bar{s}_t, \bar{s}_{t+1}) \ , \tag{22}$$

and

$$cc_t = cc^{ss}(\bar{s}_t, \bar{s}_{t+1}) \ , \tag{23}$$

where $\bar{s}_t$ represents the mean taken over all the available sample states at each time step.[3] With the above quantities defined, the final cost function (as presented in Equation 7) used to perform PILQR updates in DEALIO is prepared.

DEALIO Having resolved the incompatibility between GAIfO and PILQR above, we now describe the full proposed algorithm, DEALIO, detailed in Algorithm 1. First, we initialize a time-varying, linear Gaussian imitation controller $p(a|s)$ and a neural network discriminator D_θ (Lines 1 and 2). The imitator uses p to generate multiple state-action trajectories $\tau^i = \{(s^i, a^i)\}$ (Line 5), and these trajectories are used to fit a linear Gaussian dynamics model $P(s'|s,a)$, as shown in Equation 4 (Line 6). Next, state transitions from both the imitator and the demonstrator are used to train the discriminator D_θ (Equation 8) using the same loss function as in GAIfO (Line 7). Then, h and P are used to calculate the quadratic cost function parameters in Equation 7 (Line 8). Finally, the cost function is used to perform a PILQR update (Line 9).

[3]One alternative is to calculate the average of $C^{sa}(s_t, s_{t+1})$ and $c^{sa}(s_t, s_{t+1})$ over the available states, however, our experiments showed that calculating $C^{sa}(\bar{s}_t, \bar{s}_{t+1})$ and $c^{sa}(\bar{s}_t, \bar{s}_{t+1})$ results in better performance.

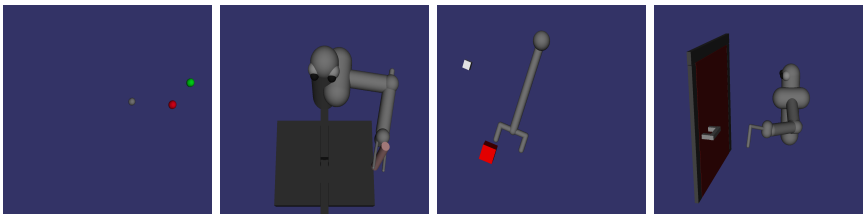
V. Experiments

In order to evaluate DEALIO, we ran several experiments using a benchmark robotics simulation environment. In particular, we sought to determine whether or not DEALIO, which integrates a model-based RL technique with the GAIfO algorithm for IfO, exhibited better data efficiency than the original GAIfO algorithm, which instead uses a model-free RL technique. We compared the data efficiency of each approach by analyzing learning curves—more data-efficient approaches exhibit steeper learning curves (i.e., they rise to a higher level of task performance using fewer samples). As we will detail below, our results do indeed confirm that DEALIO is more data-efficient than GAIfO, suggesting that integrating model-based RL approaches is a promising path forward for designing IfO approaches that are practical for real-world systems. Moreover, we also found that, even given fewer training samples, DEALIO also led to better overall task performance than GAIfO in our experimental tasks.

A. Experimental Setup

We conducted our experiments using four robotics tasks simulated using the MuJoCo physics engine [36]:

- **Disc**: This domain includes a gray point particle on a disc and two target points—one red and one green—as shown in Fig. 2(a). The agent can push the particle in the x and y directions, and the task is for the agent to first move the particle to the red target point, and then move the particle to the green target point. The state and action spaces are ten- and two-dimensional, respectively.
- **PegInsertion**: This domain includes an arm with a gripper, a peg, and a plate with a hole in it as shown in Fig. 2(b). The task is for the agent to manipulate the arm such that the peg is inserted into the hole. The state and action spaces are twenty-six- and seven-dimensional, respectively.
- **GripperPusher**: This domain includes an arm with a gripper, a white particle, and a red target point as shown in Fig. 2(c). The task is for the agent to manipulate the arm such that it reaches the particle, grabs it, and moves it to the target point. The state and action spaces are thirty-six- and four-dimensional, respectively.



(a) Disc (b) PegInsertion (c) GripperPusher (d) DoorOpening

Fig. 2. Representative screenshots of the MuJoCo domains considered in this paper.

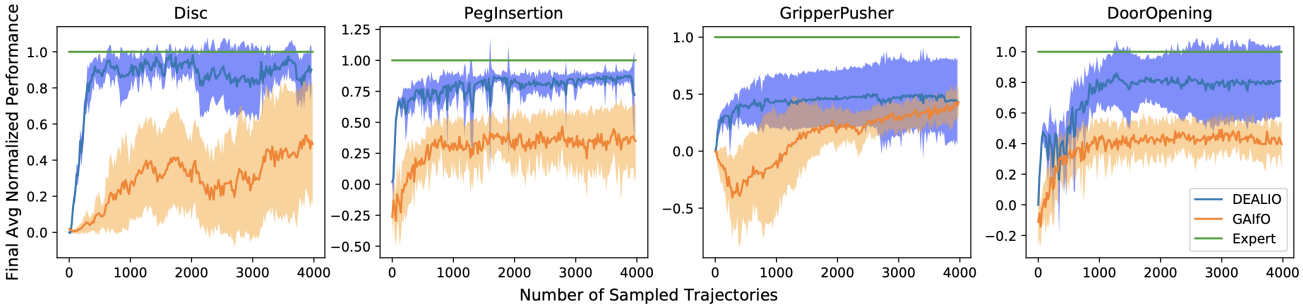


Fig. 3. Learning with DEALIO is compared against GAIfO. The plots represent the performance of the algorithms with respect to the number of trajectories sampled during the learning process. The solid lines represent the mean of the performances over 10 different training processes and the shaded areas represent the standard deviations. For comparison purposes, all the performances are scaled such that a random and the expert policy score 0.0 and 1.0, respectively.

- **DoorOpening**: This domain includes a robot arm and a door with a handle as shown in in Fig. 2(d). The task is for the agent to manipulate the arm such that it reaches the handle and opens the door. The state and action spaces are thirty-six- and six-dimensional, respectively.

In order to generate demonstration data D^e, expert demonstrators are trained for each task until convergence using PILQR with predefined task cost functions that have been previously used in related work [37], [38]. The trained policies are then used to generate 20 demonstration trajectories for each task. The performance of these expert agents is represented as the green horizontal lines in Fig. 3, where the performance metric (expected cumulative reward) is normalized to the expert's level.

B. Implementation Details

In our experiments, the neural network component of the DEALIO discriminator was modeled with two hidden layers, each with one hundred neurons. For fair comparison, the GAIfO discriminator was similarly modeled. One important hyperparameter that must be specified for both DEALIO and GAIfO is the number of trajectory samples generated by the imitator at each iteration (e.g., for DEALIO, Line 5 of Algorithm 1); we empirically determined the best value of this hyperparameter separately for each algorithm and environment. For DEALIO, we collect ten, three, twenty, and twenty trajectories (each trajectory is one hundred time steps) for each domain as ordered in Fig. 2. For GAIfO, these same parameters were set as five, five, twenty, and twenty with the same order again. Another important set of hyperparameters required by both algorithms is the number of updates to make to both the discriminator and the controller at each iteration. Again, we empirically determined the best values for these hyperparameters, and we found that ten discriminator updates per iteration and one update for the controller per iteration worked best regardless of algorithm or domain. Above, ten discriminator updates means that we divide the overall data collected at that iteration into ten batches and updated the discriminator once for each batch.

C. Results and Discussion

For each environment and algorithm, we generated the learning curves shown in Fig. 3 by computing the mean and standard deviation of the developing imitation policies over ten independent training runs. The results confirm our hypothesis that integrating a model-based RL algorithm leads to improved data efficiency: the DEALIO learning curves all exhibit a significantly steeper rise than those obtained with GAIfO. The results also show that, even over much longer training horizons, DEALIO still achieves better final performance than GAIfO. We posit that this is possible because PILQR itself combines model-free and model-based reinforcement learning algorithms such that it is both data efficient and high performing. Taken together, our results suggest that model-based RL can play an important role in bringing adversarial IfO methods to real-world scenarios.

VI. Conclusion and Future Work

In this paper, we studied the effect of integrating sample-efficient, model-based RL techniques with recent adversarial IfO algorithms. To do so, we introduced one way to perform this integration through our novel algorithm, DEALIO, and

demonstrated experimentally that (1) it exhibits significantly reduced sample complexity compared to GAIfO, and (2) it can do so seemingly without sacrificing performance.

While our work suggests that model-based RL can play an important role in improving adversarial techniques for IfO, there are several ways in which we might improve the exemplar we proposed here. One in particular concerns our use of PILQR, which is a trajectory-centric reinforcement learning algorithm, i.e., it finds a local controller, $p(a|s)$, only for a very specific task with a very specific initial state and goal. While this approach appears to have been sufficient for our experimental domains, an interesting direction for future work is to take advantage of algorithms such as Guided Policy Search (GPS) [34] to integrate general neural network policy learning using a supervised learning algorithm to mimic the combined behavior of all the learned local controllers for different initial states and goals.

REFERENCES

[1] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press Cambridge, 1998, vol. 1, no. 1.

[2] R. Zhang, F. Torabi, L. Guan, D. H. Ballard, and P. Stone, "Leveraging human guidance for deep reinforcement learning tasks," *arXiv preprint arXiv:1909.09906*, 2019.

[3] R. Zhang, F. Torabi, G. Warnell, and P. Stone, "Recent advances in leveraging human guidance for sequential decision-making tasks," *Autonomous Agents and Multi-Agent Systems*, vol. 35, no. 2, pp. 1–39, 2021.

[4] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Proceedings of the twenty-first international conference on Machine learning*. ACM, 2004, p. 1.

[5] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robot. Auton. Syst.*, vol. 57, no. 5, pp. 469–483, May 2009. [Online]. Available: http://dx.doi.org/10.1016/j.robot.2008.10.024

[6] Y. Liu, A. Gupta, P. Abbeel, and S. Levine, "Imitation from observation: Learning to imitate behaviors from raw video via context translation," *CoRR*, vol. abs/1707.03374, 2017. [Online]. Available: http://arxiv.org/abs/1707.03374

[7] F. Torabi, G. Warnell, and P. Stone, "Recent advances in imitation learning from observation," in *Proceedings of the 28th International Joint Conference on Artificial Intelligence*. AAAI Press, 2019, pp. 6325–6331.

[8] J. Merel, Y. Tassa, D. TB, S. Srinivasan, J. Lemmon, Z. Wang, G. Wayne, and N. Heess, "Learning human behaviors from motion capture by adversarial imitation," *CoRR*, vol. abs/1707.02201, 2017. [Online]. Available: http://arxiv.org/abs/1707.02201

[9] F. Torabi, G. Warnell, and P. Stone, "Generative adversarial imitation from observation," in *International Conference on Machine Learning Workshop on Imitation, Intent, and Interaction (I3)*, 2019.

[10] W. Sun, A. Vemula, B. Boots, and D. Bagnell, "Provably efficient imitation learning from observation alone," in *International Conference on Machine Learning*, 2019, pp. 6036–6045.

[11] F. Torabi, G. Warnell, and P. Stone, "Adversarial imitation learning from state-only demonstrations," in *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2019, pp. 2229–2231.

[12] J. Kober, J. A. Bagnell, and J. Peters, "Reinforcement learning in robotics: A survey," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1238–1274, 2013.

[13] J. Schulman, S. Levine, P. Moritz, M. I. Jordan, and P. Abbeel, "Trust region policy optimization," *CoRR*, vol. abs/1502.05477, 2015. [Online]. Available: http://arxiv.org/abs/1502.05477

[14] M. P. Deisenroth, G. Neumann, J. Peters *et al.*, "A survey on policy search for robotics," *Foundations and trends in Robotics*, vol. 2, no. 1-2, pp. 388–403, 2013.

[15] Y. Chebotar, K. Hausman, M. Zhang, G. Sukhatme, S. Schaal, and S. Levine, "Combining model-based and model-free updates for trajectory-centric reinforcement learning," in *ICML*. PMLR, 2017, pp. 703–711.

[16] F. Torabi, S. Geiger, G. Warnell, and P. Stone, "Sample-efficient adversarial imitation learning from observation," *arXiv preprint arXiv:1906.07374*, 2019.

[17] Y. Tassa, T. Erez, and E. Todorov, "Synthesis and stabilization of complex behaviors through online trajectory optimization," in *IROS*. IEEE, 2012, pp. 4906–4913.

[18] F. Farshidian, M. Neunert, and J. Buchli, "Learning of closed-loop motion control," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2014, pp. 1441–1446.

[19] N. Heess, G. Wayne, D. Silver, T. Lillicrap, Y. Tassa, and T. Erez, "Learning continuous control policies by stochastic value gradients," *arXiv preprint arXiv:1510.09142*, 2015.

[20] E. Theodorou, J. Buchli, and S. Schaal, "A generalized path integral control approach to reinforcement learning," *The Journal of Machine Learning Research*, vol. 11, pp. 3137–3181, 2010.

[21] F. Torabi, G. Warnell, and P. Stone, "Behavioral cloning from observation," in *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI)*, July 2018.

[22] P. Sermanet, C. Lynch, J. Hsu, and S. Levine, "Time-contrastive networks: Self-supervised learning from multi-view observation," *CoRR*, vol. abs/1704.06888, 2017.

[23] B. S. Pavse, F. Torabi, J. Hanna, G. Warnell, and P. Stone, "Ridm: Reinforced inverse dynamics modeling for learning from a single observed demonstration," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6262–6269, 2020.

[24] P. Henderson, W. Chang, P. Bacon, D. Meger, J. Pineau, and D. Precup, "Optiongan: Learning joint reward-policy options using generative adversarial inverse reinforcement learning," *CoRR*, vol. abs/1709.06683, 2017.

[25] F. Torabi, G. Warnell, and P. Stone, "Imitation learning from video by leveraging proprioception," in *International Joint Conference on Artificial Intelligence (IJCAI)*, 2019.

[26] K. Zolna, N. Rostamzadeh, Y. Bengio, S. Ahn, and P. O. Pinheiro, "Reinforced imitation learning from observations," *NeurIPS 2018 Workshop*, 2018.

[27] B. C. Stadie, P. Abbeel, and I. Sutskever, "Third-person imitation learning," *CoRR*, vol. abs/1703.01703, 2017.

[28] E. Hudson, G. Warnell, F. Torabi, and P. Stone, "Skeletal feature compensation for imitation learning with embodiment mismatch," *arXiv preprint arXiv:2104.07810*, 2021.

[29] Z. Zhu, K. Lin, B. Dai, and J. Zhou, "Off-policy imitation learning from observations," in *NeurIPS*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 12 402–12 413.

[30] C. Finn, S. Levine, and P. Abbeel, "Guided cost learning: Deep inverse optimal control via policy optimization," in *International conference on machine learning*. PMLR, 2016, pp. 49–58.

[31] 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," in *ICLR*, 2018.

[32] W. Li and E. Todorov, "Iterative linear quadratic regulator design for nonlinear biological movement systems." in *ICINCO (1)*, 2004, pp. 222–229.

[33] E. D. Sontag, *Mathematical control theory: deterministic finite dimensional systems*. Springer Science & Business Media, 2013, vol. 6.

[34] S. Levine and P. Abbeel, "Learning neural network policies with guided policy search under unknown dynamics." in *NIPS*, vol. 27. Citeseer, 2014, pp. 1071–1079.

[35] I. 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*, 2014.

[36] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2012, Vilamoura, Algarve, Portugal, October 7-12, 2012*. IEEE, 2012, pp. 5026–5033.

[37] M. Leonetti, P. Kormushev, and S. Sagratella, "Combining local and global direct derivative-free optimization for reinforcement learning," *Cybernetics and Information Technologies*, vol. 12, no. 3, pp. 53–65, 2012.

[38] W. Montgomery and S. Levine, "Guided policy search via approximate mirror descent," in *NeurIPS*, 2016, pp. 4015–4023.