Learning from Interventions:
Human-robot interaction as both explicit and implicit feedback

Jonathan Spencer[1], Sanjiban Choudhury[2], Matthew Barnes[2], Matthew Schmittle[2],
Mung Chiang[1], Peter Ramadge[1] and Siddhartha Srinivasa[2]
[1]Princeton University, [2]University of Washington
j.spencer,chiangm,ramadge@princeton.edu; schoudhury,mbarnes,schmttle,sidd@cs.uw.edu

***Abstract*—Scalable robot learning from seamless human-robot interaction is critical if robots are to solve a multitude of tasks in the real world. Current approaches to imitation learning suffer from one of two drawbacks. On the one hand, they rely solely on off-policy human demonstration, which in some cases leads to a mismatch in train-test distribution. On the other, they burden the human to label every state the learner visits, rendering it impractical in many applications. We argue that learning interactively from *expert interventions* enjoys the best of both worlds. Our key insight is that any amount of expert feedback, whether by intervention or non-intervention, provides information about the quality of the current state, the optimality of the action, or both. We formalize this as a constraint on the learner's value function, which we can efficiently learn using no regret, online learning techniques. We call our approach Expert Intervention Learning (EIL), and evaluate it on a real and simulated driving task with a human expert, where it learns collision avoidance from scratch with just a few hundred samples (about one minute) of expert control.**

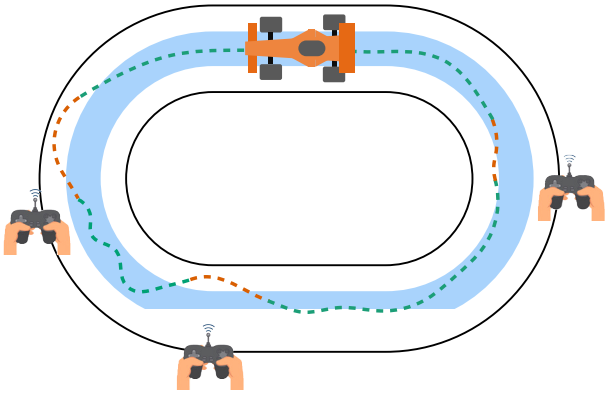


Fig. 1: In intervention learning the expert trains a learner by taking over only when needed, recovering the car and returning control to the learner. A good learner can learn a lot from when the expert does and doesn't intervene.

I. Introduction

A great many machines are designed for human control, and expert humans have mastered incredibly complex control tasks. However, as self-driving cars [1], autonomous helicopters [2], and robotic factory arms [3] have gradually developed controllers to automate simple tasks, the human's role has shifted to that of a supervisor that engages or disengages the autopilot, assuming full control when necessary. This mechanism of supervision and intervention is natural for the expert because it mimics what often occurs in human-human apprenticeship. We would like to similarly endow a robot with all of the human's expertise in a way that is both natural and efficient for the expert human instructor.

Consider the example of learning high speed rallycar driving (Fig. 1). While an expert human driver can easily demonstrate this task by driving around a track, we may require long hours of driving to cover all possible input conditions. Even so, the slightest distribution mismatch between learner and expert can result in compounding errors [4].

Another option is to interactively collect feedback [5] from the expert driver while the *learner is in control* of the car. While interactive learning addresses the distribution mismatch problem, it is impractical due to several human-robot-interaction issues. First, the learner needlessly queries the expert in states that the expert, and ideally a good learner, would never visit [6]. Secondly, the expert has no control of the how the car moves and the consequences of the feedback. This results in degraded feedback due to delayed response and subsequent over-corrections [7].

On the other hand, what if we were to give the human expert the freedom to intervene at will? How should the learner correctly interpret such interventions? Consider the scenario in Fig. 1 where the expert nominally monitors the car without any input, similar to the way they might engage with cruise control or an autopilot. As soon as the expert senses that the learner is skidding off the track, they take over, recover the car and toggle control back to the learner. This example illustrates some important truths:

- Expert interventions are natural to provide, and yet contain a lot of information.
- In many cases the expert simply wants the learner to perform well enough so the expert doesn't have to intervene.

A good learner should now learn not only how to recover in the future, but also the fact that driving near the middle of the track is much more preferable than being near the edge. Ideally the learner trajectories look something like Fig. 2; as the learner improves, the expert needs to intervene less and less.

Our key insight is that to learn a policy that is optimal everywhere you must query the expert everywhere. If you can settle for good enough, you can use implicit and explicit feedback to quickly learn a level set of the value function.

We formalize this as Expert Intervention Learning (EIL).

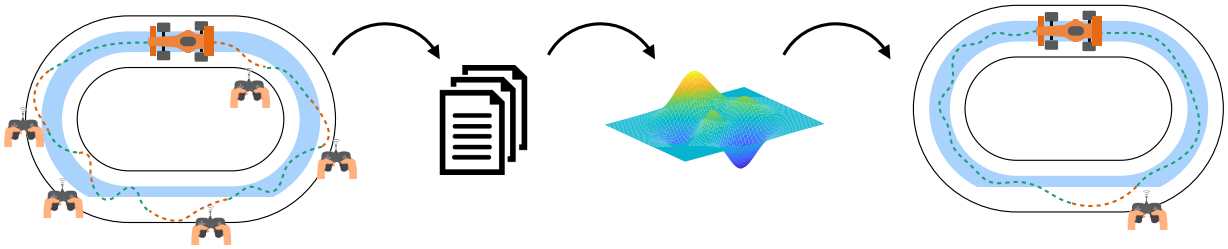


Fig. 2: Overview of EIL. It proceeds iteratively, using the current learner to collect intervention data and map said data to constraints on learner value function. It then aggregates constraints and updates the learner on all of the data

When the expert is not in control of the robot, we assume that the state-action pair is good enough. When the expert does occasionally intervene, this provides both implicit feedback about the current state being "bad" and a near-optimal trajectory to return to a "good" state. Fig. 2 shows an overview of the algorithm. At every iteration, we execute the learner, collect intervention data, aggregate it, map to constraints on the learner's action-value function, and update the learner.

Our contributions and the organization of the remainder of this paper are as follows:

1) In Section III, we formalize the notion of *good enough* performance and frame the mixed control problem in the context of existing work.
2) We introduce an algorithm in Section IV for solving mixed implicit-explicit feedback problems and show that it has desirable performance guarantees.
3) In Section V, we empirically demonstrate that our algorithm reduces the number of explicit expert interactions with the system compared to baseline methods both in simulation and using a real robot.

II. Related Work

Traditional imitation learning is known to require a large amount (quadratic in trajectory length) of expert demonstration trajectories in order to achieve expert performance [8, 9] because learner policies invariably make mistakes and deviate from expert, inducing a significantly different distribution of states from what the expert originally modeled [10, 11]. DAGGER is a foundational algorithm which addresses that problem in a provably efficient way by querying the expert online [5]. In DAGGER, the learner rolls out their current policy, then queries the expert for action labels corresponding to each state visited by the learner. The learner aggregates this set of learner-state, expert-actions with that of previous iterations, trains a new policy on the combined dataset, and iterates. This approach requires a number of expert labels that is only *linear* in trajectory length, and has been successfully applied to autonomous flight [7] and visual navigation [12, 13]. However, this remarkable improvement comes by indiscriminately querying the expert for a label at every state the learner visits, which typically happens off-line and can be both cognitively demanding and unsafe [6], inspiring alternative methods to introduce distributional diversity i.e. by injecting noise [14]. In practice, many learner samples are also redundant, leaving room for even greater sample efficiency by intelligently limiting when the expert is queried.

Several DAGGER-style algorithms employ an active learning framework, where the learner only selectively queries the expert. This decision can be based on a threshold of action-classifier confidence [15, 16, 17], distributional distance or discrepancy [18, 19], query by committee [20], or a combination of state novelty and historical error [21]. Theoretically, many of these active learning frameworks can query the expert in a batch setting off-line, however most (including DAGGER) also query the expert at execution time, resulting in a mixed control setting, where the executed trajectory switches back and forth between the human and robot on a per-sample basis and the gating (assignment of control) is done by the robot.

Robot-gated mixed control and online active learning in robotics is problematic because the *type* of samples they require are burdensome, especially in continuous control settings. Humans are sensitive to latency and timing in mixed control, and demanding sporadic samples in real-time is not only more burdensome than uninterrupted trajectories, but can also result in undesirable and unstable system dynamics [22, 6]. To this end, we instead consider human-gated mixed control, reducing (though not eliminating) the alertness burden on the expert and allowing them to determine the exact timing of handoff in a way that is more convenient and stable for them. The Human Gated DAGGER algorithm (HG-DAGGER) [22], shares this premise. They execute a human-gated mixed control trajectory, and use the human labeled portions of the trajectory (the orange portions of Fig. 2) as the online batch update for DAGGER. In addition to reducing expert burden by requiring less demanding samples, HG-DAGGER reduces the number of samples required to achieve baseline performance in driving tasks. Similar ideas have been successfully applied to reduce effort in learning from kinesthetic corrections in the manipulator control domain [23, 24, 25] or by monitoring a buffer of future robot actions in social robotic teaching [26]. We build on this by also learning from times when the expert is not in control (the green portions of Fig. 2).

Voluntary expert intervention permits the inference of deeper meaning from the timing and nature of expert feedback.

Binary feedback has been used to learn a supervisor's bias towards positive/negative and exploit that to infer implicit signaling from inaction [27], or that feedback can form an advantage function for the current policy [28]. However, a key difference between our work and most other interactive learning [29] work is that we deal with demonstrations rather than explicit positive/negative labels, instead inferring the additional positive/negative based on the timing of interventions. We limit the scope of this work to a coarse model of human intervention that provides regret guarantees, leaving for future work the intriguing questions of potential differences between multiple experts, internal human state [30], trust [31], and cooperation [32, 33] in this setting.

This work lies at the intersection of Learning from Demonstration (LfD), and interactive learning. We differ from existing corrective feedback algorithms and active learning DAGGER-style algorithms because we aim to learn from the *timing* of the expert correction in addition to copying the actions themselves. Our use of a score function enables this flexible approach, and distinguishes us from many interactive learning methods, encoding multiple modes of feedback into a set of constraints which help to quickly learn a level set dividing good and bad state-actions. Our continuous training approach introduces a scalable method of learning for many low-latency EdgeAI scenarios. This approach to learning from both implicit and explicit feedback in continuous real-time mixed-control setting is novel, efficient, easy to implement, and natural, mimicking the way that humans often teach one another.

III. Problem Formulation

We introduce a modified formulation of imitation learning where the robot tries only to be *good enough* such that the expert doesn't have to intervene. This is inspired by practicalities of domains such as self-driving or manipulation where there are several ways to accomplish the task, and the user doesn't really care to distinguish.

We model the problem as Markov Decision Process with unspecified rewards (MDP\R). Let $\mathcal{M}(\mathcal{S}, \mathcal{A}, P, T, d_0)$ be a tuple consisting of a set of states $\mathcal{S}$, a set of actions $\mathcal{A}$, an environment transition function $P : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ where $\Delta(\mathcal{S})$ is a $\mathcal{S}$-dimensional probability simplex, a fixed time horizon T, and an initial state distribution $d_0 \in \Delta(\mathcal{S})$. We restrict the learner to a class of policies Π.

We have a human expert teaching the robot via interaction. We assume the following interaction model:

1) The human deems a region of the state-action space $(s, a) \in \mathcal{G}$ to be *good enough*.
2) When the robot is in $\mathcal{G}$, the human does not intervene. The robot remains in control even though it may select actions different from what the human would have chosen.
3) As soon as the robot departs $\mathcal{G}$, the expert takes over and controls the system back to $\mathcal{G}$.

While this is a natural human-robot interaction model, it inextricably mixes state distributions induced by both expert and learner. To circumvent this problem, we modify the MDP $\mathcal{M}$. If the expert intervenes at state s_t before the end of the episode $t < T$, we mark s_t as an absorbing terminal state. Let d_π^t be the state distribution induced by following π for t steps, then $d_\pi = \frac{1}{T}\sum_{t=1}^{T} d_\pi^t$ is the average distribution of states induced by policy π. We wish to minimize the average time spent out of good states, i.e. $\mathbb{E}_{s\sim d_\pi}[1_{\{(s,\pi(s))\notin\mathcal{G}\}}]$.

However, the objective above disregards the intervention actions demonstrated by the expert. Even though such actions are off-policy, they can help regularize learning and speed up convergence. Let π_E be the expert policy. Let d_π^I be the average distribution of intervention states induced by policy π, i.e. states that the expert visits after π leaves $\mathcal{G}$. We also wish to minimize the average misclassification of intervention actions $\mathbb{E}_{s^I\sim d_\pi^I}[1_{\{\pi(s^I)\neq\pi^E(s^I)\}}]$.

We combine these objectives to define a modified imitation learning problem.

Problem 1: Find a policy that minimizes both the average time spent outside of good enough region and the average misclassification of intervention actions on its own induced distribution:

$$\min_{\pi\in\Pi} \underbrace{\mathbb{E}_{s\sim d_\pi}[1_{\{(s,\pi(s))\notin\mathcal{G}\}}]}_{\text{stay in good enough region}} + \lambda \underbrace{\mathbb{E}_{s^I\sim d_\pi^I}[1_{\{\pi(s^I)\neq\pi^E(s^I)\}}]}_{\text{learn intervention actions}}. \quad (1)$$

where λ is a tuning constant.

We note that (1) is non-convex for two reasons. First, the term inside the expectation is non-convex. Secondly, and more importantly, the induced distributions d_π and d_π^I are non-convex (even for simple convex policy classes). In the next section, we will discuss how we efficiently optimize this objective. Finally, we emphasize that Problem 1 indeed combines the best attributes of two extreme paradigms of imitation learning:

1) *Easy to provide labels as in Behavior Cloning [4]*: In fact, we argue it is even less burdensome to provide a sparse set of interventions.
2) *Correctly measures the learner induced loss as in* DAGGER *[5]*: Moreover, we do so without having to require the expert to provide labels in all the states the learner visits.

IV. Approach

We present Expert Intervention Learning (EIL), a no-regret online algorithm that learns from interventions alone. EIL builds upon and further generalizes the key insight of DAGGER [5] – *any* imitation learning objective can be reduced to an online, sequential game. Crucially, unlike DAGGER, EIL *does not* require the expert to label every state the learner enters. We show that EIL indeed enjoys the best of many worlds - it is practical, requires minimal user effort and has strong performance guarantees.

A. Modelling interventions as action-value constraints

We restrict the learner to a policy class Π of greedy policies with respect to differentiable action-value cost functions $Q_\theta(s, a)$ such that

$$\pi(s) = \arg\min_a Q_\theta(s, a) \quad (2)$$

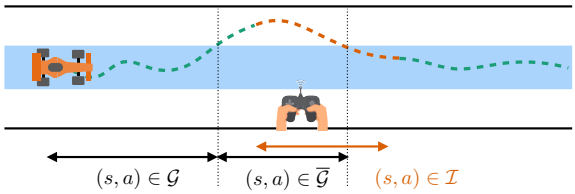


Fig. 3: Given a learner and human intervention trajectory, we can flag the data with three potential categories based on the timing of the correction - good enough state-actions $\mathcal{G}$, bad state-actions $\overline{\mathcal{G}}$ and intervention state-actions $\mathcal{I}$

Here we minimize cost, so higher $Q_\theta(s,a)$ indicates an undesirable state-action pair. Hence, we can model a good enough state-action pair $(s,a) \in \mathcal{G}$ as a threshold constraint on the action-value

$$Q_\theta(s,a) \leq B \;\; \forall (s,a) \in \mathcal{G} \tag{3}$$

where B is a scalar threshold. The precise value of B is irrelevant as it can be thought of as a way of offsetting the action-value estimator (we use $B = 0$ for our experiments).

Let an episode be represented by the learner's trajectory $\xi^L = (s_0, a_0, \cdots, s_f, a_f)$ and the subsequent expert intervention trajectory $\xi^E = (s_0^E, a_0^E, \cdots, s_f^E, a_f^E)$. Fig. 3 illustrates such an episode. Let $[\alpha, \beta) \circ \xi$ represent a *snippet* of trajectory ξ from fraction α up to β where $\alpha, \beta \in [0,1], \alpha \leq \beta$. We map snippets of ξ^L and ξ^E to three non-exclusive categories

1) *Good enough state-actions:* By not intervening during the beginning portion of ξ^L, the expert has implicitly labeled those actions as good enough, thus we label the beginning $1 - \alpha^L$ fraction of ξ^L as such.

$$(s,a) \in \mathcal{G} \;\; \forall (s,a) \in [0, 1-\alpha^L) \circ \xi^L \tag{4}$$

2) *Bad state-actions:* Upon intervention, the learner policy has clearly failed, bringing the robot into a bad state. As such, the last α_L fraction of ξ^L we label as bad states. Although the human expert chooses good *actions* we will learn to emulate, the *state* at which they take over may be undesirable. In such cases, we can choose to label the first α^E fraction of ξ^E as undesirable state-actions.

$$(s,a) \notin \overline{\mathcal{G}} \;\; \forall (s,a) \in [1-\alpha^L, 1] \circ \xi^L \cup [0, \alpha^E) \circ \xi^E \tag{5}$$

3) *Intervention state-actions:* All (s,a) pairs in ξ^E are labeled as intervention pairs.

$$(s,a) \in \mathcal{I} \;\; \forall (s,a) \in \xi^E \tag{6}$$

We discuss how to choose α_L and α_E in Section V.

We then map each of these categories to constraints on the action-value function

1) *Good enough state-actions* map to values below a threshold

$$Q_\theta(s,a) \leq B \;\; \forall (s,a) \in \mathcal{G}. \tag{7}$$

2) *Bad state-actions* map to values that are above a threshold

$$Q_\theta(s,a) > B \;\; \forall (s,a) \in \overline{\mathcal{G}}. \tag{8}$$

3) *Intervention state-actions* map to a relative action-value constraint requiring we emulate the expert in that state

$$Q_\theta(s,a) < Q_\theta(s,a') \;\; \forall (s,a) \in \mathcal{I}, a' \neq a. \tag{9}$$

Combining these constraints, we can reexpress Problem 1 as an optimization over action-value estimates

$$\begin{aligned} \min_\theta \sum_{(s,a)\in\mathcal{G}} 1_{\{Q_\theta(s,a) > B\}} + \sum_{(s,a)\in\overline{\mathcal{G}}} 1_{\{Q_\theta(s,a) \leq B\}} \\ + \lambda \sum_{(s,a)\in\mathcal{I}} \sum_{a' \neq a} 1_{\{Q_\theta(s,a) \geq Q_\theta(s,a')\}}. \end{aligned} \tag{10}$$

B. Reduction to online, convex optimization

The objective in (10) is non-convex in Q_θ. To prove performance guarantees, we need to apply convex relaxations to each of the terms[1] which we do using a convex hinge penalty.

1) *Good enough state-actions* corresponding to upper bound constraint (7) relax to

$$\ell_B^1(s,a,\theta) = \max(0, Q_\theta(s,a) - B) \;\; \forall (s,a) \in \mathcal{G}. \tag{11}$$

2) *Bad state-actions* corresponding to upper bound constraint (8) relax to

$$\ell_B^2(s,a,\theta) = \max(0, B - Q_\theta(s,a)) \;\; \forall (s,a) \in \overline{\mathcal{G}}. \tag{12}$$

3) *Intervention state-actions* correspond to a *relative* action-value constraint (9) relax to

$$\ell_C(s,a,\theta) = \sum_{a'} \max(0, Q_\theta(s,a) - Q_\theta(s,a')) \;\; \forall (s,a) \in \mathcal{I}. \tag{13}$$

We combine the first two loss functions as a bounds loss $\ell_B(\cdot) = \ell_B^1(\cdot) + \ell_B^2(\cdot)$. We can think of this as an *implicit loss* inferred from when the expert chooses to intervene. The loss $\ell_C(\cdot)$ is a classification loss. We can think of this an an *explicit loss* which uses the actual actions executed by the expert. The total loss is a weighted sum of losses $\ell(\cdot) = \ell_B(\cdot) + \lambda \ell_C(\cdot)$.

We now formally state the relaxed convex optimization problem using $d_{\pi_\theta}(s,a)$ and $d^I_{\pi_\theta}(s,a)$, the distributions induced by π_θ of nominal learner and expert intervention state, respectively. The objective is to minimize the expected loss over these induced distributions

$$\min_\theta \mathbb{E}_{(s,a)\sim d_{\pi_\theta}(s,a)} \ell_B(s,a,\theta) + \lambda \mathbb{E}_{(s,a)\sim d^I_{\pi_\theta}(s,a)} \ell_C(s,a,\theta). \tag{14}$$

Even though the losses are convex, the optimization is still non-convex because of how θ affects $d_{\pi_\theta}(s,a)$ and $d^I_{\pi_\theta}(s,a)$. We leverage a key insight from DAGGER [5] – reduce the *non-convex* imitation learning objective (14) to a *sequence of convex games*.

The game occurs between an adversary that creates loss functions and a learner that selects parameters. We define the game as follows: At round i, let θ_i be the parameters of the current learner. Let $d_i = d_{\pi_{\theta_i}}$ and $d_i^I = d^I_{\pi_{\theta_i}}$ be

[1] While we assume $Q_\theta(\cdot)$ is convex to prove regret guarantees, the update can be applied to non-convex function classes like neural networks as done in similar works [34]

Algorithm 1 Expert Intervention Learning (EIL)

Initialize data sets $\mathcal{G}$, $\overline{\mathcal{G}}$ and $\mathcal{I}$ as $\{\}$
Initialize π_1 to any policy in Π
for n = 1, ..., N **do**
  Execute learner policy π_{θ_i}.
  Get learner trajectory ξ_L and subsequent intervention trajectory ξ_E (if any).
  Aggregate (s, a) pairs to datasets $\mathcal{G}$, $\overline{\mathcal{G}}$ and $\mathcal{I}$.
  Minimize $\ell_B(s, a, \theta) + \lambda \ell_C(s, a, \theta)$ on total dataset to compute new parameter θ_{i+1}.
return best parameter from $\theta_1, \ldots, \theta_N$ on validation.

the induced distributions. The adversary chooses a convex loss $\ell_i(\theta) = \mathbb{E}_{(s,a)\sim d_i} \ell_B(s, a, \theta) + \lambda \mathbb{E}_{(s,a)\sim d_i^I} \ell_C(s, a, \theta)$. The learner proposes a parameter θ_{i+1}. The average regret is defined as

$$\gamma_N = \frac{1}{N} \sum_{i=1}^{N} \ell_i(\theta_i) - \min_{\theta} \frac{1}{N} \sum_{i=1}^{N} \ell_i(\theta). \tag{15}$$

As long as the learner chooses an update that drives regret $\gamma_N \to 0$ as $N \to \infty$ (no regret), we show in Section IV-E that the learner finds a near-optimal solution to (14). We choose Follow-the-Leader (FTL) [35]:

$$\begin{aligned} \theta_{i+1} &= \min_{\theta} \sum_{i=1}^{N} \ell_i(\theta) \\ &= \min_{\theta} \sum_{i=1}^{N} \mathbb{E}_{(s,a)\sim d_i} \ell_B(s, a, \theta) + \lambda \mathbb{E}_{(s,a)\sim d_i^I} \ell_C(s, a, \theta). \end{aligned} \tag{16}$$

FTL updates guarantee $\gamma_N = \tilde{O}(\frac{1}{N})$ for strongly convex ℓ_i, and can be realized by simply aggregating data as it is collected. Alternately, one can apply online gradient descent [36] where one need not store data.

C. Algorithm

Algorithm 1 describes EIL. At each iteration i, the learner is executed to collect trajectories ξ_L and ξ_E. These trajectories are then mapped to the 3 dataset buckets described in Section IV-A. These are then aggregated with previous datasets and the learner is trained to solve (16). The intution is that over iterations, we are building up the set of inputs the learner is likely to experience during its execution. Doing well on this dataset amounts to doing well on (14) – a concept we explore further in Section IV-E.

D. Comparison to other imitation learning frameworks

Table I places EIL with other comparable imitation learning algorithms. BC never lets the learner be in control, leading to issues such as covariate shift. DAGGER lets the learner be in control, but requires the expert to label the learner states, which the authors report to be challenging [7]. HG-DAGGER, closest to EIL, uses intervention, but only optimizes $\ell_C(\cdot)$. As we discuss in Section IV-E, this results in the learner only learning recovery behaviors rather than learning to stay in $\mathcal{G}$.

TABLE I: Different imitation learning algorithms

Algorithm	Intervention Rule	Loss Function
EIL (**ours**)	Intervene if $(s, a) \notin \mathcal{G}$	$\ell_B(\cdot) + \lambda \ell_C(\cdot)$
BC [4]	Expert in control	$\ell_C(\cdot)$
DAGGER [5]	Learner in control	$\ell_C(\cdot)^\dagger$
HG-DAGGER [22]	Intervene if $(s, a) \notin \mathcal{G}$	$\ell_C(\cdot)$

EIL gets the best of all worlds - the minimal user burden of HG-DAGGER, with DAGGER like performance guarantees.

E. Analysis

We briefly state the main results deferring the reader to supplementary materials for proofs and counter examples.

Let $i = 1, \cdots, N$ denote the rounds of the online game. Let $\theta_1, \cdots, \theta_N$ be the learner parameters in each round. Let $\ell_i(\theta)$ be the loss function for round i. We build on [5] to show that any no-regret algorithm can achieve near-optimal performance.

Theorem 1: Let $\ell_i(\theta) = \mathbb{E}_{(s,a)\sim d_{\pi_{\theta_i}}} \ell(s, a, \theta)$. Let $\epsilon_N = \min_\theta \frac{1}{N} \sum_{i=1}^{N} \ell_i(\theta)$ be the loss of the best parameter in hindsight after N iterations. Let γ_N be the average regret of $\theta_{1:N}$. There exists a $\theta \in \theta_{1:N}$ s.t.

$$\mathbb{E}_{(s,a)\sim d_{\pi_\theta}} [\ell(s, a, \theta)] \leq \epsilon_N + \gamma_N \tag{17}$$

Theorem 1 is a simple, but powerful generalization because it extends for any loss function $\ell(s, a, \theta)$ and induced distribution $d_{\pi_\theta}(s, a)$. Because our objective is separable, we can also use this to prove a set of corollaries for variants of the EIL algorithm.

Consider the case where we use only the implicit bounds loss, i.e. only a flag to indicate whether (s, a) is good enough.

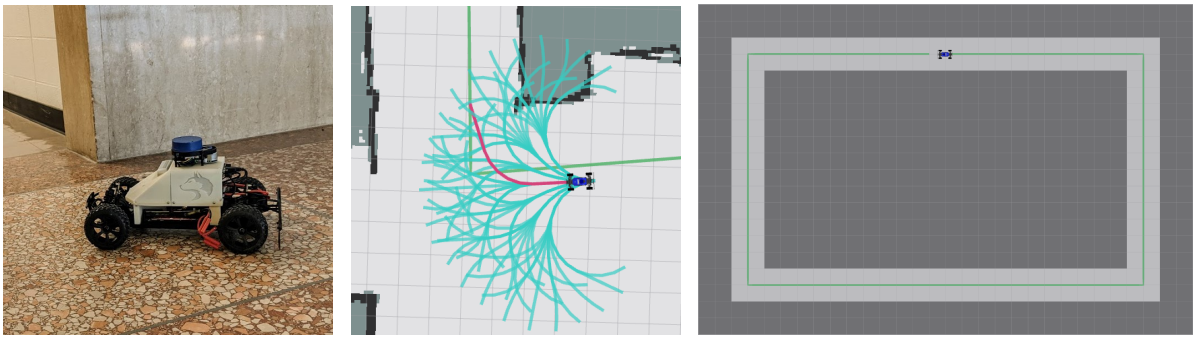
Corollary 1: Let $\ell_i(\theta) = \mathbb{E}_{(s,a)\sim d_{\pi_{\theta_i}}} \ell_B(s, a, \theta)$. Let ϵ_N^B and γ_N^B be the best loss in hindsight and average regret respectively. $\exists \theta \in \theta_{1:N}$ s.t. $\mathbb{E}_{(s,a)\sim d_{\pi_\theta}} [\ell_B(s, a, \theta)] \leq \epsilon_N^B + \gamma_N^B$, i.e. we can use EIL to learn a near-optimal policy with *as little as Boolean feedback*, e.g. from only e-stop disengagements as long as we employ FTL.

Now consider the case where we use only the intervention loss, i.e. the HG-DAGGER [22] algorithm.

Corollary 2: Let $\ell_i(\theta) = \mathbb{E}_{(s,a)\sim d^I_{\pi_{\theta_i}}} \ell_C(s, a, \theta)$. Let ϵ_N^I and γ_N^I be the best loss in hindsight and average regret respectively. $\exists \theta \in \theta_{1:N}$ s.t. $\mathbb{E}_{(s,a)\sim d^I_{\pi_\theta}} [\ell_C(s, a, \theta)] \leq \epsilon_N^I + \gamma_N^I$, i.e. we can learn to be near-optimal w.r.t mimicking intervention recovery. However, *such a policy may perform arbitrarily poorly when it comes to avoiding intervention in the first place*, staying inside $(s, a) \in \mathcal{G}$. We bolster this with a counter example (refer to supplementary) and empirical observations.

Finally, EIL combines both bound and intervention losses

Corollary 3: Let $\ell_i(\theta) = \mathbb{E}_{(s,a)\sim d_{\pi_{\theta_i}}} \ell_B(s, a, \theta) + \lambda \mathbb{E}_{(s,a)\sim d^I_{\pi_{\theta_i}}} \ell_C(s, a, \theta)$. Let ϵ_N and γ_N be the best loss in hindsight and average regret respectively. $\exists \theta \in \theta_{1:N}$ s.t. $\mathbb{E}_{(s,a)\sim d_{\pi_\theta}} [\ell_B(s, a, \theta)] + \lambda \mathbb{E}_{(s,a)\sim d^I_{\pi_\theta}} [\ell_C(s, a, \theta)] \leq \epsilon_N + \gamma_N$, i.e. it performs near-optimally on the combination of the induced bounds and intervention loss.



(a) MuSHR driving platform (b) MPC discrete action set (blue) and chosen action (red) (c) Generic rectangular track mission

Fig. 4: The Multi-agent System for non-Holonomic Racing (MuSHR) robot is a driving platform that uses model predictive control over a discrete library of possible trajectories (b), but permits takeover from human supervisor. Our mission is to complete collision free laps around a rectangular track (c).

V. Experiments

We test EIL in a robot driving task where the goal is to track a coarse reference path in a way that is good enough (i.e. collision free, smooth) so the expert need not intervene. The reference path (Fig. 4b,c) may have sharp turns or pass through obstacles - hence we wish to learn a controller than can track it appropriately. We focus on repeated training episodes at specific locations (right turn, straight hallway) to benchmark how quickly the robot learns that specific skill from scratch.

A. Experimental Setup

Robot Agent – All robot experiments (sim and real) use the Multi-agent System for non-Holonomic Racing (MuSHR) driving and simulation platform [37], a 1/10 scale car equipped with: short range lidar, RGBD camera, IMU and NVIDIA Jetson Nano (Fig. 4a). We use lidar to manually create a prior map, which we localize against at run-time. Because of the onboard EdgeAI computer, both safety override controls and policy updates are low latency, making it a very natural candidate for training with EIL (Fig. 4b). Sim and real use an identical model predictive controller (MPC) where:

- State $s_t \in \mathcal{S}$ – localized pose and velocity.
- Low-level Control u_t – steering angle ϕ_t and acceleration.
- Action space $\mathcal{A}$ – a fixed library of 64 motion primitives (see Fig. 4b). Each action $a^{(i)}$ is a sequence of pre-defined control and resulting predicted states $a_t^{(i)} = (u_{1:H}^{(i)}, s_{t:t+H}^{(i)})$.
- Feature function $\boldsymbol{f} : s, a \rightarrow \mathbb{R}^d$ – for action primitive a we compute the average over states in the primitive horizon $s_{t:t+H}$ for each of: 0-1 boundary violation, absolute curvature to next step, distance to nearest obstacle, and distance to goal path. (We also include unity bias feature.)
- Score function $Q_\theta(s,a) = \theta^T \boldsymbol{f}(s,a)$ – linear in features.
- Policy π_θ – greedy cost minimizer (red path in Fig. 4b) $\pi_\theta(s) = \arg\min_{a \in \mathcal{A}} \; Q_\theta(s,a)$.
- Trajectory $\xi = \{(s_t, a_t, u_t)\}_{t=1}^T$ – a sequence of at most T state-action-control samples of either/both robot or human.

At timestep t, the controller evaluates the policy $a_t = \pi_\theta(s_t)$, executes the first control u_t of a_t, and the process repeats.

Expert and Hyperparameters – In simulation, the expert uses a set of manually tuned optimal feature weights θ_E, over the same score and policy classes as the learner, $\pi_E(s) = \arg\min_{a \in \mathcal{A}} \; Q_{\theta_E}(s,a)$. We also set an intervention threshold B_E and decide whether to intervene or cede control by continuously scoring the current action w.r.t. Q_E. If the learner action exceeds B_E, the expert supplies subsequent actions until the score drops back below the threshold. In choosing B_E, the expert effectively defines the size of the good enough region, and we discuss how its choice affects performance in Section V-B. Although a human expert has no such precise internal Q_E, θ_E, or B_E, they provide feedback in a similar way, by intervening and supplying a sequence of nominal or intervention controls $(u_t, \ldots, u_{t+N})$ based on when they perceive the robot as acting poorly or to avoid collision. We project the sequence of expert controls back to the discrete primitive action space $\mathcal{A}$ by minimizing the Fréchet distance [38] between the sequence of states during expert control and the projected states visited for each action in the library $\arg\min_{a \in \mathcal{A}} F((s_{t:t+N}|u_{t:t+N}), (s_{t:t+N}|a))$.

The optimal choice of hyperparameters α_L and α_E depends on properties of both the robot and the human expert, such as MPC horizon and reaction time, though poorly chosen values smoothly degrade performance to be similar to HG-DAGGER. For both sim and real, the MPC horizon is $H = 15$ steps ($\sim$ 3m), so we choose α_L so that the last H learner steps before intervention are flagged as bad. In our context, MPC horizon length (rather than expert reaction time) is the dominant factor in choosing α_L since the robot typically lands in a bad state after contemplating it for H steps, and H is usually much longer than the expert reaction time. α_E depends on how quickly the expert recovers and returns to $\mathcal{G}$, but in this context $\alpha_E = 0$ works fine for a skilled expert. Choosing λ very large or very small utilizes only half the available feedback, which slows learning. For our environment, a fixed $\lambda = 1$ worked well, though future work could explore adaptively tuning λ as the dataset grows. Likewise, using multiple or sub-optimal experts would certainly require more care, and a variant of

the Bayesian approach of [27] might present a way to infer appropriate values of α_L. We set $B = 0$ as it is just a way to coarsely rank states, and it absorbs into the bias.

Experiment Setup, Baselines, and Evaluation – Each policy improvement iteration n as denoted in Algorithm 1 can either be done by gathering a batch of samples under the current policy, or in a fully online manner, updating the policy after each time-step. For the sake of repeatability, we opt for the batch setting, further breaking the mission (Fig. 4c) into two portions trained independently: a straight hallway segment and a sharp corner. The hallway is interesting because it is repetitive, i.e. it should learn with few samples. The corner is interesting because driving through the tip of the corner has only a modest effect on the feature function but is catastrophic for safety. All algorithms are initialized with the same set of (very bad) parameters and given one full example trajectory. For every improvement iteration n, we initialize the car at roughly the same location, generate a short trajectory (50-70 steps, 10-15m), aggregate, improve, and iterate.

We choose as baselines the algorithms listed in Table I, and evaluate based on 1) the total number of samples in the environment as well as 2) the number of samples supplied by the expert. For Behavioral Cloning (BC) and DAGGER, those two numbers will be identical, since the expert labels every state, though DAGGER has slightly fewer total samples over the same number of iterations due to learner policy crashing and terminating early. We hold the number of iterations constant, so EIL and HG-DAGGER have substantially fewer expert samples since the expert intervenes only when necessary.

In simulation, we judge success according to action suboptimality on a fixed validation set $\mathcal{D}_T$, $E_{a\sim\pi_L, s\sim\mathcal{D}_T}[Q_E(s, a_L) - Q_E(s, a_E)]$ as a consistent, low-variance benchmark. With a human, the standard of "good enough" is very flexible. Our expert is instructed to intervene *as consistently as possible* based on a) collision avoidance and b) "jerkiness" $\sum_t |\dot{\phi}_t|$. We measure ultimate success by the number of samples required before the policy consistently executes collision-free trajectories and the jerkiness of the converged policy.

B. Experimental Results

MuSHR Simulation Results – From our simulation experiments, we make the following observations:

Observation 1: EIL outperforms all algorithms on all datasets both in number of expert samples and total number of environment samples.

In Fig. 5 we see that EIL achieves the highest performance, while HG-DAGGER performs comparably to DAGGER in terms of total samples, but outperforms DAGGER in the number of expert samples since it doesn't query the expert on the segments of learner control. HG-DAGGER amounts to an implementation of EIL with no reliance on implicit feedback, (remove ℓ_B term or make λ very large), which indicates that the improvement in sample efficiency is largely due to the added benefit of implicit feedback. Surprisingly, the hallway proves to be the more challenging task in simulation, because many states are repetitive, and it takes a longer time

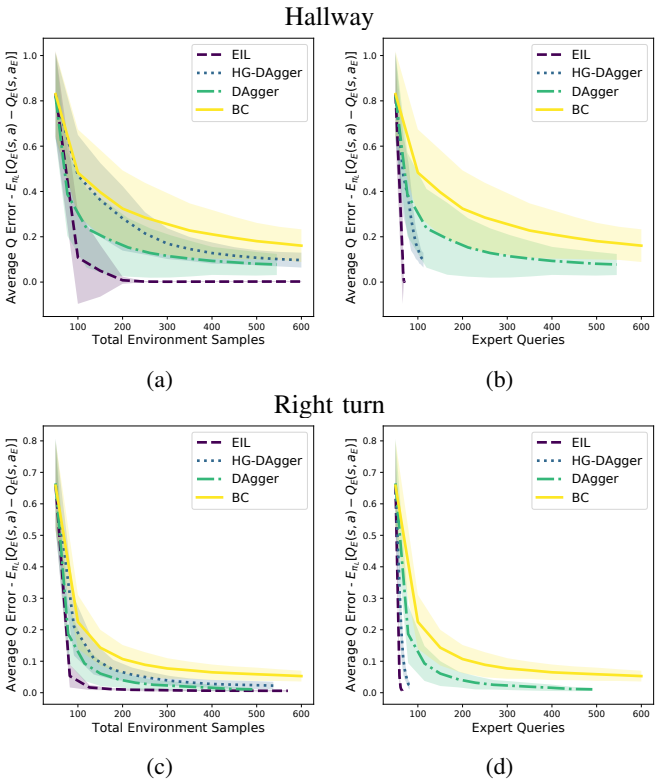


Fig. 5: Simulation performance of EIL compared to baselines (see Table I) in a straight hallway scenario (a,b) and the right hand turn segment (c,d) pictured in Fig. 4b.

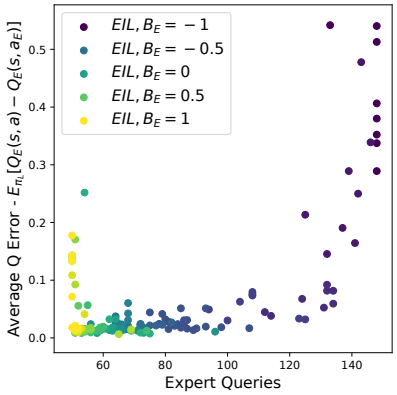


Fig. 6: Performance in driving sim after 200 time-steps for different intervention thresholds B_E. Varying B_E confirms intuitions that an overly aggressive expert is sample inefficient, while an overly passive expert fails to provide useful feedback.

to discover the key states for shaping optimal performance. On the physical system, however, added localization error makes precisely clearing the sharp corner the more challenging task.

Observation 2: There exists an optimal level of intervention which minimizes both learner error and expert burden. (Fig. 6)

The hand-crafted π_{θ_E} expert in simulation lets us precisely track learner performance as a function of the expert intervention style, parameterized here by the intervention threshold

TABLE II: Samples required to achieve a zero collision policy for right turn, jerkiness ($\sum_t |\dot{\phi}_t|$) and mean obstacle proximity (m) of converged policies after 24 iterations of $T = 70$ steps.

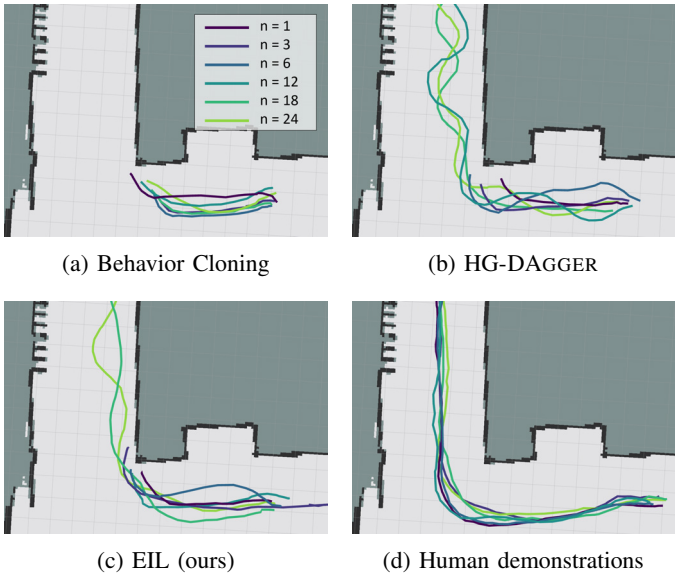
Alg.	# Expert	# Total	Jerkiness	Obst. Prox
BC [4]	> 1680	> 1680	0.0090	0.341
EIL (**ours**)	311	1120	0.0135	0.396
HG-DAGGER	223	560	0.0217	0.437
Human	-	-	0.0114	0.675

B_E. In Fig. 6 we explore the range between an aggressively intervening expert ($B_E = -1$) and an overly passive expert ($B_E = 1$). Reducing $B_E \to -\infty$ shrinks the set of good states to nothing and mimics behavioral cloning in requiring that the expert always be in control. The aggressive expert prevents the learner from making mistakes and visiting diverse states that would speed learning, and thus burdens the expert by requiring more samples. On the other hand, by raising the cost threshold $B_E \to \infty$, a passive expert almost never intervenes, giving the agent too much leeway and the agent fails to learn, receiving neither correction nor demonstration. Although a human expert cannot make such precise intervention style adjustments, this gives a useful heuristic: In implementing EIL (and perhaps in life) some intervention is helpful, but too much correction will both exhaust the supervisor and slow the learning process, and too little will rob the learner of meaningful expertise.

MuSHR Robot with Human Expert – Our experiments with the physical robot and the human expert corroborate our first observation from simulation and add the following observation:

Observation 3: Learning only from recovery trajectories can harm performance.

Table II shows the number of samples required for the learner to achieve a policy that consistently avoids collision in the right turn scenario. After 24 expert demo trajectories, BC never achieves collision avoidance, demonstrating the difficulty of this task. HG-DAGGER learns collision avoidance with somewhat fewer samples than EIL, but the converged policy (after 24 iterations) is still undesirable, making jerky weaving maneuvers seen in Fig. 7b. This is unsurprising, because the HG-DAGGER dataset is imbalanced, largely comprised of jerky explicit "recovery" policy samples supplied by the expert. EIL benefits from the reinforcement of the implicit approval of the smooth parts of its steering and the implicit disapproval of any jerky samples as it swerves towards a wall prior to intervention, and the inclusion of $\alpha_E > 0$ can teach the robot to avoid the initial jerky expert states. In Table II and Fig. 7 we see that EIL quickly learns to both avoid collision and smoothly track the reference in a "good enough" way for the expert not to intervene.



(a) Behavior Cloning (b) HG-DAGGER

(c) EIL (ours) (d) Human demonstrations

Fig. 7: Recorded rollouts of π_{θ_m} show the policy improvement as training progresses for each algorithm.

VI. Discussion

Learning from demonstration in a way that is both sample efficient and easy to implement is challenging since many techniques ask human experts to perform burdensome off-line labeling [5, 14, 21]. *Expert Intervention Learning* introduces a novel way to train robots that is natural to implement, provides theoretical guarantees, and demonstrates strong performance in practice. EIL exploits both implicit and explicit feedback from corrective demonstrations, the benefit of which we see clearly in the policies produced by the real robot experiment. Relying solely on recovery actions creates an imbalanced dataset, biased towards actions exhibiting undesirable jerky behaviors, whereas the incorporation of coarse implicit feedback gives a more balanced sample set, producing both safe and desirable policies (Fig. 7). Our simulation results also suggest a trade-off for the expert in deciding how good is "good enough" and how strictly to enforce it (Fig. 6). With regard to interventions, if the expert can afford to be patient, then less is more.

Our approach is successful when the expert is consistent and we are satisfied with simply achieving a performance threshold. We posited that EIL is natural and un-burdensome, but supervision still requires alertness, and we plan to do a user study to see just how much our claim holds, and what happens when the expert is less consistent or sub-optimal. Our coarse goodness threshold was key for harnessing implicit feedback, but an interesting avenue of future work is to incorporate a multi-task learning framework, and thus learn from multiple experts with perhaps different thresholds and deduce which objectives they are biased towards [27]. Finally, we also plan to look into the way an expert may change and adapt intervention technique as learning progresses, extending theory of mind into interventions, which can benefit the wider body of human-robot interaction.

Acknowledgment

This work was (partially) funded by the DARPA Dispersed Computing program, NIH R01 (#R01EB019335), NSF CPS (#1544797), NSF NRI (#1637748), the Office of Naval Research, RCTA, Amazon, and Honda Research Institute USA.

References

[1] A. Sadat, M. Ren, A. Pokrovsky, Y.-C. Lin, E. Yumer, and R. Urtasun, "Jointly learnable behavior and trajectory planning for self-driving vehicles," *arXiv preprint arXiv:1910.04586*, 2019.

[2] S. Choudhury, V. Dugar, S. Maeta, B. MacAllister, S. Arora, D. Althoff, and S. Scherer, "High performance and safe flight of full-scale helicopters from takeoff to landing with an ensemble of planners," *Journal of Field Robotics (JFR)*, 2019.

[3] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen, "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection," *The International Journal of Robotics Research (IJRR)*, 2018.

[4] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," in *Advances in Neural Information Processing Systems (NeurIPS)*, 1989.

[5] S. Ross, G. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics (AIStats)*, 2011.

[6] M. Laskey, C. Chuck, J. Lee, J. Mahler, S. Krishnan, K. Jamieson, A. Dragan, and K. Goldberg, "Comparing human-centric and robot-centric sampling for robot deep learning from demonstrations," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2017.

[7] S. Ross, N. Melik-Barkhudarov, K. S. Shankar, A. Wendel, D. Dey, J. A. Bagnell, and M. Hebert, "Learning monocular reactive UAV control in cluttered natural environments," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2013.

[8] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Proceedings of the twenty-first International Conference on Machine learning (ICML)*, 2004.

[9] H. Daumé III, J. Langford, and D. Marcu, "Search-based structured prediction," *Machine Learning Journal (MLJ)*, 2009.

[10] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and Autonomous Systems*, 2009.

[11] T. Osa, J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, and J. Peters, "An algorithmic perspective on imitation learning," *Foundations and Trends in Robotics*, 2018.

[12] S. Gupta, J. Davidson, S. Levine, R. Sukthankar, and J. Malik, "Cognitive mapping and planning for visual navigation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[13] B. Kim, A. Farahmand, J. Pineau, and D. Precup, "Learning from limited demonstrations," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2013.

[14] M. Laskey, J. Lee, W. Hsieh, R. Liaw, J. Mahler, R. Fox, and K. Goldberg, "Iterative noise injection for scalable imitation learning," *arXiv preprint arXiv:1703.09327*, 2017.

[15] S. Chernova and M. Veloso, "Interactive policy learning through confidence-based autonomy," *Journal of Artificial Intelligence Research*, 2009.

[16] K. Menda, K. R. Driggs-Campbell, and M. J. Kochenderfer, "EnsembleDAgger: A Bayesian Approach to Safe Imitation Learning," *arXiv preprint arXiv:1807.08364*, 2018.

[17] D. H. Grollman and O. C. Jenkins, "Dogged learning for robots," in *Proceedings 2007 IEEE International Conference on Robotics and Automation (ICRA)*, 2007.

[18] B. Kim and J. Pineau, "Maximum mean discrepancy imitation learning." in *Robotics: Science and Systems (RSS)*, 2013.

[19] B. Packard and S. Ontañón, "Policies for active learning from demonstration," in *2017 AAAI Spring Symposium Series*, 2017.

[20] K. Judah, A. P. Fern, and T. G. Dietterich, "Active imitation learning via reduction to iid active learning," in *2012 AAAI Fall Symposium Series*, 2012.

[21] M. Laskey, S. Staszak, W. Y.-S. Hsieh, J. Mahler, F. T. Pokorny, A. D. Dragan, and K. Goldberg, "SHIV: Reducing supervisor burden in dagger using support vectors for efficient learning from demonstrations in high dimensional state spaces," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016.

[22] M. Kelly, C. Sidrane, K. Driggs-Campbell, and M. J. Kochenderfer, "Hg-dagger: Interactive imitation learning with human experts," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019.

[23] A. Bajcsy, D. P. Losey, M. K. O'Malley, and A. D. Dragan, "Learning robot objectives from physical human interaction," in *Proceedings of the 1st Annual Conference on Robot Learning (CoRL)*. PMLR, 2017.

[24] A. Jain, B. Wojcik, T. Joachims, and A. Saxena, "Learning trajectory preferences for manipulators via iterative improvement," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2013.

[25] A. Bajcsy, D. P. Losey, M. K. O'Malley, and A. D. Dragan, "Learning from physical human corrections, one feature at a time," in *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2018.

[26] E. Senft, P. Baxter, and T. Belpaeme, "Human-guided learning of social action selection for robot-assisted therapy," in *Machine Learning for Interactive Systems*, 2015.

[27] R. Loftin, B. Peng, J. MacGlashan, M. L. Littman, M. E. Taylor, J. Huang, and D. L. Roberts, "Learning behaviors via human-delivered discrete feedback: modeling implicit feedback strategies to speed up learning," *Autonomous Agents and Multi-Agent Systems*, 2016.

[28] J. MacGlashan, M. K. Ho, R. Loftin, B. Peng, G. Wang, D. L. Roberts, M. E. Taylor, and M. L. Littman, "Interactive learning from policy-dependent human feedback,"

in *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 2017.

[29] S. Amershi, M. Cakmak, W. B. Knox, and T. Kulesza, "Power to the people: The role of humans in interactive machine learning," *AI Magazine*, vol. 35, pp. 105–120, 2014.

[30] D. Sadigh, S. S. Sastry, S. A. Seshia, and A. Dragan, "Information gathering actions over human internal state," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016.

[31] M. Chen, S. Nikolaidis, H. Soh, D. Hsu, and S. Srinivasa, "Planning with trust for human-robot collaboration," in *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2018.

[32] D. Hadfield-Menell, S. J. Russell, P. Abbeel, and A. Dragan, "Cooperative inverse reinforcement learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

[33] J. F. Fisac, M. A. Gates, J. B. Hamrick, C. Liu, D. Hadfield-Menell, M. Palaniappan, D. Malik, S. S. Sastry, T. L. Griffiths, and A. D. Dragan, "Pragmatic-pedagogic value alignment," *Robotics Research*, p. 49–57, Nov 2019.

[34] W. Sun, A. Venkatraman, G. J. Gordon, B. Boots, and J. A. Bagnell, "Deeply AggreVaTeD: Differentiable Imitation Learning for Sequential Prediction," in *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 2017.

[35] S. Shalev-Shwartz, "Online learning and online convex optimization," *Foundations and Trends in Machine Learning*, vol. 4, no. 2, pp. 107–194, 2012.

[36] M. Zinkevich, "Online convex programming and generalized infinitesimal gradient ascent," in *Proceedings of the 20th International Conference on Machine Learning (ICML)*, 2003.

[37] S. S. Srinivasa, P. Lancaster, J. Michalove, M. Schmittle, C. Summers, M. Rockett, J. R. Smith, S. Choudhury, C. Mavrogiannis, and F. Sadeghi, "MuSHR: A Low-Cost, Open-Source Robotic Racecar for Education and Research," *arXiv preprint arXiv:1908.08031*, 2019.

[38] H. Alt and M. Godau, "Computing the Fréchet distance between two polygonal curves," *International Journal of Computational Geometry & Applications*, vol. 5, pp. 75–91, 1995.