# Uncertainty-Aware Policy Sampling and Mixing for Safe Interactive Imitation Learning

Manfred Diaz[1,2] Thomas Fevens[2] and Liam Paull[1]

***Abstract*— Teaching robots how to execute tasks through demonstrations is appealing since it sidesteps the need to explicitly specify a reward function. However, posing imitation learning as a simple supervised learning problem suffers from the well-known problem of *distributional shift* - the teacher will only demonstrate the optimal trajectory and therefore the learner is unable to recover if it deviates even slightly from this trajectory since it has no training data for this case. This problem has been overcome in the literature by some element of *interactivity* in the learning process - usually be somehow interleaving the execution of the learner and the teacher so that the teacher can demonstrate to the learner also how to recover from mistakes. In this paper, we consider the cases where the robot has the potential to do harm, and therefore *safety* must be imposed at every step in the learning process. We show that uncertainty is an appropriate measure of safety and that both the mixing of the policies and the data sampling procedure benefit from considering the uncertainty of both the learner and the teacher. Our method, uncertainty-aware policy sampling and mixing (UPMS), is used to teach an agent to drive down a lane with less safety violations and less queries to the teacher than state-of-the-art methods.**

## I. INTRODUCTION

The use of machine learning algorithms in robotics is becoming mainstream with applications in a rich and an increasing number of domains, such as controlling self-driving cars [1], aiding in diagnosis and healthcare treatments [2], among many other areas. One issue which separates many robotics scenarios from the standard machine learning problem is *safety*: the physical agent should not cause harm or injury during either the learning or the deployment phase.

In an Imitation Learning (IL) setting, an agent learns a policy from demonstrations from a teacher. One issue with this approach is that if the learner deviates slightly from the teacher's optimal trajectory at test time, then it has no training data to help it recover and it quickly diverges. Broadly speaking in machine learning, this problem is known as *distributional shift* since the training and testing data are sampled from different distributions.

One common approach to overcome the problem of distributional shift is through Interactive Imitation Learning (IIL) which allow the learner's policy to influence the distribution of the training samples [3]–[6]. These methods selectively allow the learner to execute its is own predictions (as would happen at test time) during training to gradually improve the robustness of the final policy. Typically this is achieved by sampling some time $t$ uniformly from $[0, T]$ where $T$ is the horizon of the task, and then allowing the learning to execute it's policy for time $[0, t]$ and then allow the expert to *recover* from time $[t + 1, T]$. However, these methods generally do not explicitly consider the issue of safety for the learner in the sense that they assume that the expert will always be able to recover to the optimal trajectory from any state that results from the learner execution until $t$ (e.g., [5] Sec. 2.2 and [4] Sec. 2.2).

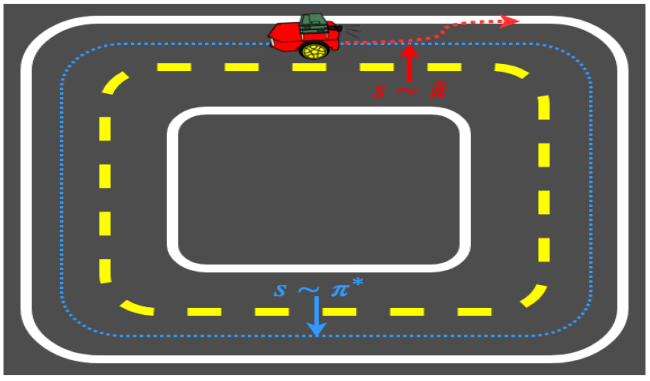



Fig. 1: In an imitation learning approach to driving, a mistake made by a learned policy $\hat{\pi}$ induces a distributional shift from the teacher $\pi^*$ since the teacher only demonstrates the optimal policy. As a result, the learner quickly diverges. Our algorithm, UPMS, is capable of correcting for this problem and guarantee expert-bounded safety throughout the learning process.

This assumption is likely to be incorrect in some scenarios. For example, consider the stochasticity inherent in the learner's non-converged policy in earlier stages of training for tasks with large horizons. A random sampling of $t$ could allow the untrained learner's random policy to execute until catastrophic failure before the expert is even given the chance to recover. For example, an autonomous vehicle may have already collided with an obstacle. In short, a strategy such as this may disproportionately give execution time to either the learner's policy execution or the exploration mechanism that disregards the global status of the learning procedure.

In this work, we propose an IIL approach that attempts to address these safety concerns inherent in fielding and deploying real physical robotic systems. This is achieved through a reformulation of the IIL problem to make it *uncertainty aware* - the policy mixing and time horizon sampling strategies explicitly estimate and account for the uncertainties in both the learner and the expert.

In summary we claim the following advantages of our proposed method over the state of the art:

- It requires less queries to the teacher to learn a policy
- It guarantees *expert-bounded* safety during learning

We demonstrate these claims on the reproducible Duckietown environment [7].

[1] Mila, Department of Computer Science and Operations Research, University of Montréal, Montréal, Canada. {manfred.ramon.diaz.cabrera,liam.paull}@umontreal.ca

[2]Department of Computer Science and Software Engineering, Concordia University, Montréal, Canada. fevens@cse.concordia.ca

## II. BACKGROUND AND RELATED WORKS

The imitation learning problem can be framed as a Markov Decision Process (MDP). A MDP is a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$ where $\mathcal{S}$ is a finite set of states $s \in \mathcal{S}$, $\mathcal{A}$ is a finite set of actions $a \in \mathcal{A}$, $\mathcal{P}$ is the state transition probability (system dynamics), $\mathcal{R}$ is the reward function, and $\gamma \in [0, 1]$ is the discount factor [8]. A policy $\pi : \mathcal{S} \rightarrow p(\mathcal{A}|\mathcal{S})$ is a stochastic decision rule that specifies an agent's behavior through the horizon (duration) $T$ of the task.

IL is the learning process between two agents, the *learner* and the *teacher*, where we derive a learner control policy $\hat{\pi}$ using a finite set of traces of the teacher policy $\pi^*$ execution $\tau = \{(s_0, a_0^*), \dots, (s_N, a_N^*)\}$. The goal is to minimize the discrepancy between $\hat{\pi}$ and the $\pi^*$ control values at each state $s \in S$ under a similarity measure $e(\hat{\pi}(s), \pi^*(s))$.

IL have the advantage that it is not necessary to specify a reward function $\mathcal{R}$. IL algorithms can be sub-divided into two families: *Direct Policy Derivation* and *Indirect Policy Derivation*. Traditionally, Direct Policy Derivation (DPD) methods have attempted to directly recover the expert's policy $\pi^*$ from $\tau$. In contrast, Indirect Policy Derivation (IPD) algorithms try to learn the reward function from $\tau$ via *Inverse Reinforcement Learning* in order to later solve the control problem with Reinforcement Learning [9].

We will present a DPD algorithm algorithm which is able to guarantee safety up to the limit of the teacher and learn policies with minimal teacher queries.

### A. Imitation via Supervised Learning

Supervised Learning (SL) describes a procedure where a dataset $\mathcal{D} = \{(x_0, y_0), ..., (x_N, y_N)\}$ containing pairs of input ($x_i \in \mathcal{X}$) and output ($y_i \in \mathcal{Y}$) vectors is used to find a hypothesis (or classifier) $\theta : \mathcal{X} \mapsto \mathcal{Y}$ over a hypothesis space $\Theta$.

Posing IL as SL is at a first glance straightforward. We can simply impose the following equivalences: $\tau \equiv \mathcal{D}$, $\mathcal{S} \equiv \mathcal{X}$, $\mathcal{A} \equiv \mathcal{Y}$, and reduce the policy derivation to finding a parameterization $\theta \in \Theta$ that represents $\pi \in \Pi$. Thus, the policy parameterization $\pi_\theta$ can be obtained by finding the hypothesis that minimizes the expected empirical error over the traces:

$$\hat{\pi}_\theta = \underset{\theta \in \Theta}{\operatorname{argmin}} \, \mathbb{E}_{s \sim \tau}[e(\pi_\theta(s), \pi^*(s))] \tag{1}$$

*1) The Problem of Distributional Shift:* Although quite simple to implement, solving IL via SL violates the *i.i.d assumptions*: (a) that each samples in a dataset are independent and that (b) the training and test sets originate from the same distribution. The application of SL to a sequential control problem satisfies neither of these requirements. The violation of (a) is a result of the sequential nature of the decision making problem. The assumption (b) is violated with high likelihood because at some time step during the execution of the learned policy $\hat{\pi}$, the predicted action will deviate from the ones seen during training (for example due to the stochastic nature of the control input or the fact that $\hat{\pi} \neq \pi^*$). In this case the system could end up in a region of the state space not represented by the training set. The result is that when the learner makes a mistake as a result of its own predictions, the test distribution is shifted. This phenomenon is usually referred to in the statistical learning literature as *distributional shift* or *exposure bias* and it is common to structured prediction problems [10].

Consider specifically the autonomous driving scenario depicted in Fig. 1. If the teacher is always executing a (near) optimal policy, then, if the learner deviates from the optimal path it will have no data in $\mathcal{D}$ to help guide it back to the optimal path and will quickly diverge.

### B. Interactive Imitation Learning

A common strategy to overcome the problem of distributional shift in IL is to incorporate some level of *interactivity* between the learner and the teacher during training.

Four tools to enable this in different ways have become prevalent and will combine to form the baseline algorithm on which we will build: *policy mixing, on-policy data aggregation, explicit exploration* and *online learning*.

*1) Policy Mixing:* In general form, the problem of policy mixing (initially introduced in [3]) is defined as:

**Definition 1.** ***Policy Mixing*** *Given two policies $\pi^a$ and $\pi^b$ and mixing coefficients $\alpha_i^a$ and $\alpha_i^b$ with $\alpha_i^a + \alpha_i^b = 1$ then the mixture policy at iteration $i$ is denoted by:*

$$\textbf{mix} \quad \pi_i \leftarrow \alpha_i^a \pi^a + \alpha_i^b \pi^b$$

*where $\pi_i$ is equal to $\pi^a$ with probability $\alpha_i^a$ and equal to $\pi^b$ with probability $\alpha_i^a$.*

Mixing $\hat{\pi}$ with the $\pi^*$ during training induces an *implicit exploration* step through erroneous predictions and exposes the learner to a distribution different from the one obtained by executing $\pi^*$ only.

*2) On-Policy Data Aggregation:* The interwoven execution of $\hat{\pi}$ and $\pi^*$ during training makes it possible to build an aggregate dataset $\mathcal{D}$ at each time step that is stochastically composed of both policies' predictions and can be used for training. This technique is referred to as *dataset aggregation* [5].

$$\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_i \ , \forall \ i \tag{2}$$

*3) Explicit Exploration:* As learning progresses, $\hat{\pi}$ will converge towards $\pi^*$. Implicit exploration achieved through policy mixing and dataset aggregation has improved robustness somewhat, but can be further achieved by selectively allowing the learner to *explore* [4], [11].

For instance, this is achieved in the AggreVaTe algorithm by interleaving through policy mixing with an *explicit exploration* policy $\pi_{\mathcal{E}}$ [4]. A time instant $t$ over the task's horizon is sampled, the learner's policy is executed up to time $t$, the exploration policy is executed at time $t$ and then the teacher executes. The hope is that this strategy will allow the system to visit potentially unexplored parts of its state space and learn from the expert's corrective behaviour. One issue with this approach is that it is impossible to provide guarantees on the safety of the learner and exploratory policies, particularly if the task horizon is large.

**Alg. 1** INTERACTIVE IMITATION LEARNING

**Require:** $\pi^*$: teacher's policy, $N$: iterations, $p^{1:N}$: decay rate

$\mathcal{D} \leftarrow \emptyset$, $\pi_1 \in \Pi$
**for** $i \leftarrow 1...N$ **do**
  **mix** $\pi_i \leftarrow p^i\pi^* + (1-p^i)\hat{\pi}_{i-1}$ ▷ Sec. II-B.1
  $\mathcal{D}_i \leftarrow$ RANDOMSAMPLING$(\pi_i)$
  $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_i$ ▷ Sec. II-B.2
  $\hat{\pi}_i = \underset{\pi \in \Pi}{\operatorname{argmin}} \mathbb{E}_{s \sim d_{\pi_i}}[e(\pi(s), \pi^*(s))]$ ▷ Sec. II-B.4
**return** $\hat{\pi}_N$

**Alg. 2** RANDOM SAMPLING

**Require:** $\pi^*$: teacher's policy, $\pi_{\mathcal{E}}$: exploration policy, $T$: planning horizon, $M$: number of trajectories (= 1 if running on real robot)

**procedure** RANDOMSAMPLING$(\pi_i)$
  $\mathcal{D} \leftarrow \emptyset$
  **for** $m \leftarrow 1..M$ **do**
    **sample** $t \sim U(1, T)$
    **execute** $\pi_i$ from $1..t-1$
    **explore** $\pi_{\mathcal{E}}(s_t)$ ▷ Sec. II-B.3
    **execute** $\pi^*$ from $t+1...T$
    $\mathcal{D} \leftarrow \mathcal{D} \cup \{(s_{t+1}, a^*_{t+1}), \ldots, (s_t, a^*_T)\}$
  **return** $D$

*4) Batch Online Learning:* Finally, the system is only really interactive if the learner is able to update its model based on data while it is being collected. The online learning process can be expressed as:

$$\hat{\pi}_i = \underset{\pi \in \Pi}{\operatorname{argmin}} \mathbb{E}_{s \sim d_{\pi_i}}[e(\pi(s), \pi^*(s))] \tag{3}$$

where $d_{\pi_i}$ is the set of datapoints generated by following the policy mixture $\pi_i$. The minimization procedure happens when $\mathcal{D}_i$ batches of training samples are aggregated to the task dataset $\mathcal{D}$ . This *batched* online learning setting is slightly different to the traditional online machine learning where samples are provided sequentially. Although it is difficult to guarantee convergence to global optima with state-of-the-art parameterization (e.g. Deep Neural Networks), DPD algorithms with batched online learning still generally work well in practice [12], [13].

Alg. 1 and 2 summarize the baseline algorithm including policy mixing, dataset aggregation, explicit exploration and online learning. Hereafter, this will be referred to as General Interactive Imitation Learning (GIIL).

### C. Safe Imitation Learning

Central to the problem of safety in decision-making is the ergodicity assumption [14] that states that there are no unrecoverable regions of the state space. This assumption is impractical and dangerous in real physical systems [15]. A formal method to ensure safety on MDPs involves identifying reachability sets, or regions of the system state space which are guaranteed to be safe [16], [17]. But it is unclear whether reachability sets are sufficient for ensuring safety in an MDP [15], [18], [19]. However, the IL setup provides a different context since the expert demonstrates an optimally-safe behaviour under the assumptions of a rational and non-adversarial expert. For example, the Confident Execution Framework (CEF) [20] requires the parameterization of the learner's policy to be a confidence-aware classifier to ensure an adjustable autonomy level during training using a threshold over the classifier confidence. CEF has been extended to consider a multi-thresholded approach as a single threshold was deemed insufficient for classifiers of higher complexity [21]. Other derivations of CEF explore metrics for establishing boundaries between the teacher and the learner agent based on a single-threshold approach [22]–[24]. For instance, Maximum Mean Discrepancy-IL [22] computes an upper bound over the discrepancy between the learner and the teacher policy predictions to mix policies. SafeDAgger [23] trains a safety binary classifier that helps to execute the learner's policy only when the classifier predicts a safe scenario. Finally, DropoutDAgger [24] extends SafeDAgger by estimating the uncertainty of the learner using Monte Carlo Dropout [25]. One shortcoming is that confidence-based learner-driven systems have the potential to be exposed to overconfident predictions or under or over estimation of safety and do not consistently offer the teacher an opportunity to prevent these issues. To overcome this issue, the method we propose here integrates the notion of the expert's uncertainty into the the IIL framework.

The Confidence-Based Autonomy Framework (CBAF) [26] gives the most comprehensive attempt at safe IL as it is the only method that allows expert's interventions during learning. However, CBAF's approach requires the computation of the model's decision boundaries instead of its uncertainty, an unpractical approach with current machine learning models.

## III. PROPOSED METHOD

The core element of recent literature in safe IIL centers around being about to estimate *confidence* or its inverse, *uncertainty*. For generality, we will assume that there is some function $\mathcal{U}_\pi : \mathcal{S} \rightarrow \mathbb{R}_{\geq 0}$ that evaluates the uncertainty over the actions that will be produced by a given policy $\pi$ at state $s$.

We assume that our IIL algorithm has access to a measure of the uncertainty of the teacher, $\mathcal{U}_{\pi^*}$, and a measure of the learner's uncertainty $\mathcal{U}_{\hat{\pi}}$. The IIL can be constrained to be "safe" by limiting the amount of uncertainty in the policy that is tolerated during learning. In general, this constraint could act over the teacher's uncertainty, the learner's uncertainty (as done in DropoutDAgger [24]), or, as we propose, a mixture of both.

Another way to interpret our objective is that we are looking to derive a policy mixing (Sec. III-A) and sampling (Sec. III-B) strategy that allows our learning agent to safely collect samples about all of the states that it might ever visit at test time while always guaranteeing that the teacher can return the system to the region of states that would be visited if only ever following the teacher's policy. In theoretical terms, we are enforcing that the space of policies in the batch online learning optimization (3) is constrained to only the safe ones.

### A. Uncertainty-Aware Rational Policy Mixing

The policy mixing problem is defined in Problem 1. We desire the following properties in our policy mixing:

1) *Consistency* or the ability to maintain probabilistic consistent coefficients $\alpha_{\pi_p} + \alpha_{\pi_s} = 1$.
2) *Rationality* or the ability to asymptotically select the best policy under uncertainty.
3) *Neutrality* or the ability to assign equal probabilities to equally uncertain policies.
4) *Impossibility* or the ability to detect an impossible decision under uncertainty.

Previous methods such as MMD-IL [22], SafeDAgger [27] or DropoutDAgger [24] violate *rationality*, *neutrality*, and *impossibility* since the uncertainty of only one of the two possible policies is considered. Let's assume that we have one preferred policy $\pi_p$ (usually the learner) and one secondary policy $\pi_s$

In this case, the uncertainty-aware policy mixing between teacher and learner is achieved via Uncertainty-Aware *Preferential* Policy Mixing as defined below.

**Definition 2.** ***Uncertainty-Aware Preferential Policy Mixing (UA-PPM)*** *The choice of $\alpha_p$: $f(\mathcal{U}_p)$, $0 \geq \alpha_p \geq 1$ expresses preference for policy $\pi_p$ through the following two rules:*

$$\begin{aligned} \lim_{\mathcal{U}_p \to 0^+} \alpha_p \pi_p + (1 - \alpha_p)\pi_s = \pi_p \\ \lim_{\mathcal{U}_p \to \infty} \alpha_p \pi_p + (1 - \alpha_p)\pi_s = \pi_s \end{aligned} \tag{4}$$

*regardless of the uncertainty of the secondary policy $\mathcal{U}_s$.*

A Safe IIL system should not express in its design preference for either $\mathcal{U}_{\hat{\pi}}$ or $\mathcal{U}_{\pi^*}$, instead it should establish a cooperation boundary as it selects the policy that is more confident to execute (e.g, CBAF [21]).

To express the neutrality in the selection, we can use $\alpha_{\pi_p} = f(U_{\pi_p}),\ \alpha_{\pi_s} = f(U_{\pi_s})$ as mixing coefficients which yields:

$$\textbf{mix}\ \pi_{RM} \leftarrow \alpha_{\pi_p}\pi_p + \alpha_{\pi_s}\pi_s \tag{5}$$

Our four desired properties can now be expressed in terms of the behaviour of the mixture through the asymptotic limits of $\mathcal{U}_p$ and $\mathcal{U}_s$:

$$\begin{aligned} \alpha_{\pi_p} + \alpha_{\pi_q} &= 1 && \text{(consistency)} \\ \lim_{\mathcal{U}_p \to 0,\ \mathcal{U}_s \to \infty} \pi_{RM} &= \pi_p && \text{(rationality)} \\ \lim_{\mathcal{U}_p \to \infty,\ \mathcal{U}_s \to 0} \pi_{RM} &= \pi_s && \text{(rationality)} \\ \lim_{\mathcal{U}_p \to 0,\ \mathcal{U}_s \to 0} \pi_{RM} &= \frac{1}{2}\pi_p + \frac{1}{2}\pi_s && \text{(neutrality)} \\ \lim_{\mathcal{U}_p \to \infty,\ \mathcal{U}_s \to \infty} \pi_{RM} &: \text{undefined} && \text{(impossibility)} \end{aligned} \tag{6}$$

Under these conditions, a rational mixture of policies $\pi_{RM}$ guarantees that a more certain policy has a higher probability of being selected *(rationality)*. For those cases when no policy can be rationally selected *(impossibility)* an undefined scenario is created and left open to the system designer. We also removed with this design any preference for either $\pi^*$ or $\hat{\pi}$ uncertainties *(neutrality)*.

**Alg. 3** UNCERTAINTY-AWARE POLICY MIXING AND SAMPLING

**Require:** $\pi^*$, $N$, $T$, $U_{\pi^*}$, $U_{\hat{\pi}_{i-1}}$, $f$
1: $\mathcal{D} \leftarrow \emptyset$, $\pi_1 \in \Pi$
2: **for** $i \leftarrow 1...N$ **do**
3: $\quad \alpha_{\pi^*} \leftarrow f(U_{\pi^*}),\ \alpha_{\hat{\pi}_{i-1}} \leftarrow f(U_{\hat{\pi}_{i-1}})$
4: $\quad$ **mix** $\pi_{RM} \leftarrow \alpha_{\pi^*}\pi^* + \alpha_{\hat{\pi}_{i-1}}\hat{\pi}_{i-1}$
5: $\quad \mathcal{D}_i \leftarrow$ RATIONALSAMPLING$(\pi_{RM})$
6: $\quad \mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_i$
7: $\quad \hat{\pi}_i = \underset{\pi \in \Pi}{\operatorname{argmin}}\, \mathbb{E}_{s \sim d_{\pi_{RM}}}[e(\pi(s), \pi^*(s))]$
8: **return** $\hat{\pi}_N$

**Theorem 1.** *The hyperbolic tangent function:*

$$\alpha_\pi = f(\mathcal{U}_\pi) \triangleq 1 - \tanh(\mathcal{U}_\pi) \tag{7}$$

*satisfies the four desired properties of consistency, rationality, neutrality, and impossibility as defined in* (6)*.*

*Proof.* Please see [28]. □

We refer to this process as Uncertainty-Aware Rational Policy Mixing (UA-RPM).

### B. Uncertainty-Aware Rational Sampling

As shown in Alg. 2, standard methods for IIL generate trajectories by: sampling a time within the task horizon: $t \sim U(1, T)$, executing the learner policy for time $t' = 1..t-1$, executing an exploration policy at time $t$, and then executing the teacher policy for time $t' = t+1..T$. This strategy is problematic from a safety standpoint since there is no guarantee of safety during the first $t$ steps of execution before the teacher is allowed to take over.

We propose to mitigate this issue by using policy mixing *in the sampling process* to have a much more fine-grained control of which policy executes at each time step of a rollout. We use $\alpha_{\pi_{RM}} = f(\mathcal{U}_{\pi_{RM}})$, the uncertainty of the mixture produced in (5) to rationally balance the execution of $\pi_{RM}$ and the exploratory policy $\pi_\mathcal{E}$ in a way such that $\pi_\mathcal{E}$ is only executed in low uncertainty contexts. We achieve this with Uncertainty-Aware Preferential Policy Mixing (UA-PPM) between $\pi_{RM}$ and $\pi_\mathcal{E}$ expressing a preference for $\pi_{RM}$.

This is the principled theoretical way to guarantee safety, however, it assumes correct estimation of the learner uncertainty $\mathcal{U}_{\hat{\pi}_i}$, which is actually quite difficult to estimate in practice. In the case that it is underestimated, $\pi_\mathcal{E}$ would be selected more frequently, impeding the expert's intervention through the mixture. Thus, we re-formulate an exploration strategy $\pi'_\mathcal{E}$ through another UA-PPM that expresses preference over $\pi^*$:

$$\textbf{mix}\ \pi'_\mathcal{E} \leftarrow \alpha_{\pi^*}\pi^* + (1 - \alpha_{\pi^*})\pi_\mathcal{E} \tag{8}$$

Then, the Uncertainty-Aware Rational Sampling (UA-RS) mechanism can be re-stated as:

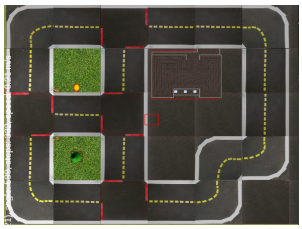
$$\textbf{mix}\ \pi_{RS} \leftarrow \alpha_{\pi_{RM}}\pi_{RM} + (1 - \alpha_{\pi_{RM}})\pi'_\mathcal{E} \tag{9}$$

Alg. 3 and 4 describe the full algorithm for Uncertainty-Aware Policy Mixing and Sampling (UPMS) , a method that by introducing minimal changes to the policy mixing and sampling mechanisms improves *expert-bounded* safety in GIIL.

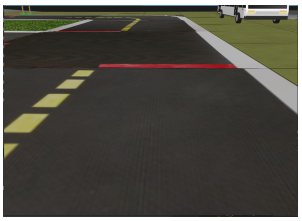
**Alg. 4** UNCERTAINTY-AWARE RATIONAL SAMPLING

**Require:** $\pi^*$, $\pi_{\mathcal{E}}$, $T$, $M$ (similar to Alg. 2)

**procedure** RATIONALSAMPLING($\pi_i$)
  $\mathcal{D} \leftarrow \emptyset$
  **for** $m \leftarrow 1..M$ **do**
    **for** $t \leftarrow 1..T$ **do**
      $\alpha_{\pi_i} \leftarrow f(\mathcal{U}_{\pi_i})$, $\alpha_{\pi^*} \leftarrow f(\mathcal{U}_{\pi^*})$
      **mix** $\pi_{\mathcal{E}'} \leftarrow \alpha_{\pi^*}\pi^* + (1 - \alpha_{\pi^*})\pi_{\mathcal{E}}$
      **mix** $\pi_{RS} \leftarrow \alpha_{\pi_i}\pi_{RM} + (1 - \alpha_{\pi_{RM}})\pi_{\mathcal{E}'}$
      **execute** $\pi_{RS}$
      $\mathcal{D} \leftarrow \mathcal{D} \cup (s_t, a_t^*)$
  **return** $\mathcal{D}$

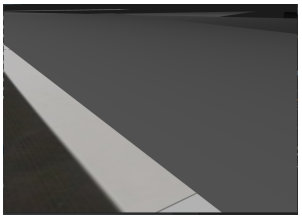


(a) Duckietown OpenAI Gym Environments: UdeM 1 map.



(b) Agent in-lane position.

(c) Agent out-of-lane position.



(d) Agent close to world bounds.

Fig. 2: The Duckietown OpenAI Gym Environment [29]

## IV. EXPERIMENTAL SETUP

This section presents the application of UPMS to the problem of lane following. Using a single monocular camera as input and a simulated environment, the experiments also develop a comparative analysis of the safety, performance and learning efficiency of UPMS against other state-of-the-art IIL algorithms.

### A. Duckietown OpenAI Gym Environments

The Duckietown OpenAI Gym Environment (DOGE) [29] is an OpenAI Gym [30] platform for autonomous driving that can be used either in simulation or in the real Duckietown environment [7].

To ensure fairness across all the algorithms evaluated, it is necessary to precisely define the protocol for gathering experts demonstrations and interactively training and test IIL algorithms for the lane following task. Across all the execution, the horizon of this task was estimated to 512 simulation steps which are approximately equivalent to one lap around the outer lane of this map. Each algorithm was trained for ten episodes with pre-computed starting points that were chosen at random and that represented a correct position and orientation in the lane. The demonstrations were collected from a human expert through the DOGE joystick interface to simplify the training procedure. The samples were obtained from driving around the outer lane of the *UdeM-1* map (Figure 2a) either in simulation or in the real environment.

In these experiments and across each algorithm, the learner's policy was parametric with a convolutional neural network model architecture that is based on the Residual Networks architecture proposed in [31].

In the experiments configuration presented here, a fully interactive learning procedure is implemented for all evaluated algorithms with the following rules:

1) The expert policy $\pi^*$ has control over the first training episode.
2) Starting the second episode, the expert intervenes ($\mathcal{U}_{\pi^*} = 0$) only as per the requirement of the algorithms.

It is important to state that the requirements of the algorithms in (2) refer to either querying requirements or input of teacher's uncertainty given to each algorithm accordingly.

Furthermore, to reduce stochasticity in the environment and improve reproducibility of the experiments, all the random generators were initialized using a pre-computed seed that was kept constant across algorithms in the same iteration (e.g., $seed = 1234$ for all algorithms in iteration 1).

The learning samples were fed to the learner's policy parameterization through online batch learning [4]. This method helps alleviate the noise in the gradient used to optimize this model. Regarding the optimization method implemented, it was empirically determined that *AdaGrad* [32] with an initial learning rate of $0.001$ obtained the best results during online training.

## V. RESULTS AND DISCUSSION

In our evaluation we compare our algorithm against several baselines: Supervised (only teacher), DAgger [5], AggreVaTe [4], and DropoutDAgger [24]. The uncertainty threshold (predictive variance) $\sigma^2$ was empirically computed for the task and set to $\sigma^2 = 0.1$. This selection also impacted the mixture coefficients in UPMS. To ensure fairness and to make the threshold point close to the saturation point of $1 - \tanh(\sigma^2) \approx 3$ the uncertainty values were multiplied by 30 units.

Also, to isolate the impact of the changes introduced by UPMS, the evaluation procedure included three variations to the original definition of UPMS: UPMS-NE performs no explicit exploration, UPMS-SL tries to address a possible *data starvation* problem in UPMS by aggregating $\hat{\pi}(s)$ predictions to the training dataset, and UPMS-NE-SL accounted for a combination of the strategies explained above.

Table I presents the qualitative results obtained during the training phase of each algorithm on the lane following task in DOGE. *Penalties* are given by the environment when the learning system invades the opposite lane or moves contrary to the lane direction. When the robot completely leaves the road, this is deemed a catastrophic failure and is listed as a *safety violation*. That penalties and out of bounds events constitute the primary measure of safety for an IIL algorithm in our evaluation. The analysis of the results presented in Table I demonstrates that uncertainty-aware IIL algorithms (DropoutDAgger and UPMS variants) offer better safety

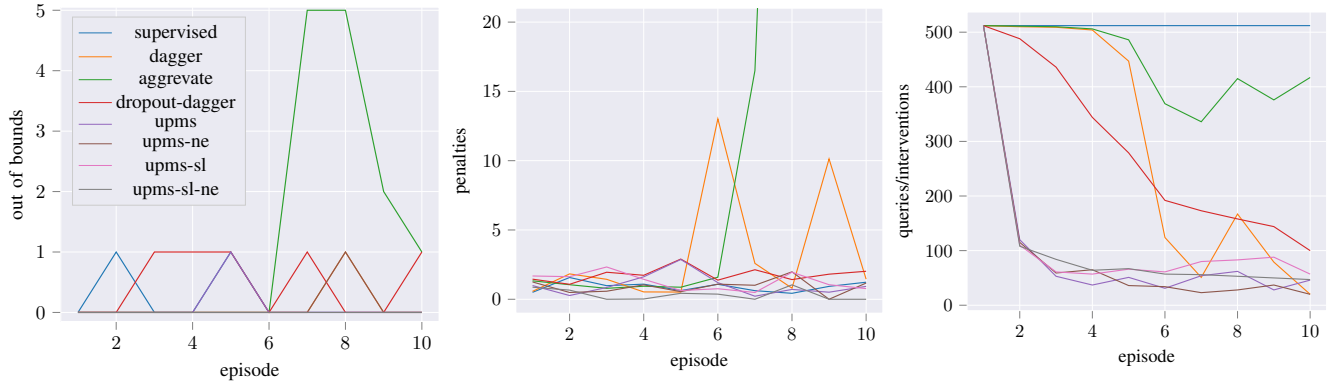



Fig. 3: **(a)** Number of out of bounds events over training episodes, **(b)** Penalty values obtained by each algorithm over training episodes, **(c)** Number of teacher interventions over training episodes. (Note: legend in (a) applies to (b) and (c))

| **Algorithms** | Penalties Received | Safety Violations | Inter-ventions |
|---|---|---|---|
| Supervised (baseline) | $9.06 \pm 0.36$ | 2 | 5120 |
| DAgger ($\alpha_1 = 0.99$) [5] | $32.88 \pm 4.47$ | 1 | 2924 |
| AggreVaTe ($\alpha_1 = 0.99$) [4] | $252.63 \pm 43.58$ | 13 | 4440 |
| DropoutDAgger [24] ($\sigma^2 > 0.1$) | $17.87 \pm 0.51$ | 5 | 2826 |
| UPMS ($\alpha = 1 - \tanh(30 \cdot \sigma^2)$) | $10.18 \pm 0.78$ | 1 | 994 |
| UPMS-NE (no exploration) | $9.18 \pm 0.54$ | 1 | 931 |
| UPMS-SL (self-learning) | $12.74 \pm 0.62$ | 0 | 1175 |
| UPMS-NE-SL | $3.53 \pm 0.34$ | 0 | 1100 |

TABLE I: Training penalties, safety violations, and number of queries to the expert (interventions) originated from the training phase of each IIL algorithm in DOGE.

guarantees than traditional IIL learning systems while requiring fewer interventions/queries to the expert's policy, a result that reaffirms the findings in literature [24]. Furthermore, UPMS and its variants significantly exceed DropoutDAgger (and baselines) performance across all the evaluation parameters over the full training steps (5120). As discussed before, one of the main drawbacks of DropoutDAgger - and any learner-preferred policy mixing strategy in general - is that it does not entirely address the safety issues in the learning system. Failures in the uncertainty estimation function or misspecification of the uncertainty threshold may provoke the overconfident execution of the learner's policy. This phenomenon is reflected by the penalties and out of bounds quantities for this method entry in Table I. In aggregate, UPMS is able to achieve safer training (both in terms of safety violations and penalties) with few interventions from the teacher.

It is also essential to corroborate the consistency of these results from the perspective of how the learning system evolves over training episodes. Fig. 3c shows per-episode values of the number of queries/interventions for all algorithms over the 10 episodes they were executed.

An unintended effect of incorporating uncertainty estimation in the IIL framework is the reduction of the number of queries (or interventions) of the expert policy. This effect was first observed in [23] and posteriorly corroborated in [24]. As displayed in Figure 3c, the number of queries/interventions in DropoutDAgger –which surpasses SafeDAgger [23] in this regard– decreases linearly with the number of episodes. The uncertainty-based shared control strategy proposed here (UPMS and variants) has significantly improved this behaviour. All of the UPMS variants have dramatically faster convergence in terms of the number of required interventions, demonstrating that learning is happening much more efficiently. Also important to note, is that, due to our rational sampling strategy, the teacher is permitted to intervene whenever deemed necessary. As a result, *all* of the interventions are at times when the learner needs aid. This is contrast to the uniform random sampling startegy under which the teacher might want to intervene but not be allowed because it is prior to the sampled time $t$, or might be forced to intervene even though it isn't necessary.

Figures 3a, 3b show the number of *out of bounds* events and the values of the penalties received in each training episode respectively. We note a performance drop in DAgger and AggreVaTe towards the end of training empirically solidifies the analysis that hypothesis these algorithms suffer from safety issues. This situation arises when the mixture of policies becomes preferential towards the learner's non-converged policy, making the learning system harder to control by the teachers's input. Uncertainty-aware IIL algorithms are less affected by this phenomenon. DropoutDAgger still presents some issues in this regard that may be directly related to the selection of the uncertainty threshold, issues in the uncertainty estimation method, or that the shared control strategy is deficient. UPMS and its variants perform consistently in this aspect as the values for penalties remain approximately constant and low through training and very few *out of bounds* events occur.

For a quantitative demonstration of the system in action on the real robot please refer to the media attachment.

## VI. CONCLUSION

We have presented a method for safe interactive imitation learning. It enforces safety by leveraging uncertainty in both the policy mixing and the sampling stages. Our results show that the agent is able to efficiently learn a policy with few required teacher interventions while satisfying safety constraints throughout the learning process.

## REFERENCES


[1] M. Bojarski, D. D. Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao, and K. Zieba, "End to End Learning for Self-Driving Cars," *arxiv preprint arXiv:1604.07316v1*, 2016.

[2] E. Choi, A. Schuetz, W. F. Stewart, and J. Sun, "Medical Concept Representation Learning from Electronic Health Records and its Application on Heart Failure Prediction," *CoRR*, 2017. [Online]. Available: https://arxiv.org/pdf/1602.03686.pdf

[3] S. Ross and J. A. Bagnell, "Efficient Reductions for Imitation Learning," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 2010, pp. 661—-668. [Online]. Available: https://www.ri.cmu.edu/pub_files/2010/5/Ross-AIStats10-paper.pdf

[4] ——, "Reinforcement and Imitation Learning via Interactive No-Regret Learning," 2014. [Online]. Available: https://arxiv.org/pdf/1406.5979.pdf

[5] S. Ross, G. J. Gordon, and J. A. 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.*, 2011, pp. 627–635. [Online]. Available: https://www.cs.cmu.edu/ sross1/publications/Ross-AIStats11-NoRegret.pdf

[6] W. Sun, A. Venkatraman, G. J. Gordon, B. Boots, and J. A. Bagnell, "Deeply AggreVaTeD: Differentiable Imitation Learning for Sequential Prediction," *Proceedings of the 34th International Conference on Machine Learning*, vol. 70, pp. 3309–3318, 2017. [Online]. Available: http://proceedings.mlr.press/v70/sun17d/sun17d.pdf

[7] L. Paull, J. Tani, H. Ahn, J. Alonso-Mora, L. Carlone, M. Cap, Y. F. Chen, C. Choi, J. Dusek, Y. Fang, D. Hoehener, S. Liu, M. Novitzky, I. F. Okuyama, J. Pazis, G. Rosman, V. Varricchio, H. Wang, D. Yershov, H. Zhao, M. Benjamin, C. Carr, M. Zuber, S. Karaman, E. Frazzoli, D. D. Vecchio, D. Rus, J. How, J. Leonard, and A. Censi, "Duckietown: An open, inexpensive and flexible platform for autonomy education and research," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, may 2017, pp. 1497–1504.

[8] W. Sun, J. A. Bagnell, and B. Boots, "Truncated Horizon Policy Search: Deep Combination of Reinforcement and Imitation," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=ryUlhzWCZ

[9] S. Russell, "Learning Agents for Uncertain Environments (Extended Abstract)," in *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, ser. COLT' 98. New York, NY, USA: ACM, 1998, pp. 101–103.

[10] Joaquin Quinonero-Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence, *Dataset Shift in Machine Learning*, T. G. Dietterich and M. I. Jordan, Eds. Cambridge, MA, USA: MIT Press, 2009. [Online]. Available: http://doi.wiley.com/10.1111/j.1467-985X.2009.00624_10.x

[11] M. Laskey, J. Lee, R. Fox, A. Dragan, and K. Goldberg, "DART: Noise Injection for Robust Imitation Learning," Tech. Rep. [Online]. Available: https://arxiv.org/pdf/1703.09327.pdf

[12] Y. Pan, C.-A. Cheng, K. Saigol, K. Lee, X. Yan, E. Theodorou, and B. Boots, "Agile Autonomous Driving using End-to-End Deep Imitation Learning," *arXiv preprint arXiv:1709.07174*, 2018. [Online]. Available: https://arxiv.org/pdf/1709.07174.pdf

[13] S. Ross, N. Melik-Barkhudarov, K. Shaurya Shankar, A. Wendel, D. Dey, J. A. Bagnell, and M. Hebert, "Learning Monocular Reactive UAV Control in Cluttered Natural Environments," Tech. Rep. [Online]. Available: https://arxiv.org/pdf/1211.1690.pdf

[14] J. García and F. Fernández, "A Comprehensive Survey on Safe Reinforcement Learning," *Journal of Machine Learning Research*, vol. 16, pp. 1437–1480, 2015. [Online]. Available: http://jmlr.org/papers/volume16/garcia15a/garcia15a.pdf

[15] T. M. Moldovan and P. Abbeel, "Safe Exploration in Markov Decision Processes," in *Proceedings of the 29th International Coference on International Conference on Machine Learning*, ser. ICML'12. USA: Omnipress, 2012, pp. 1451–1458. [Online]. Available: https://arxiv.org/pdf/1205.4810.pdf http://dl.acm.org/citation.cfm?id=3042573.3042759

[16] J. H. Gillulay and C. J. Tomlin, "Guaranteed safe online learning of a bounded system," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, sep 2011, pp. 2979–2984.

[17] J. F. Fisac, A. K. Akametalu, M. N. Zeilinger, S. Kaynama, J. Gillula, and C. J. Tomlin, "A General Safety Framework for Learning-Based Control in Uncertain Robotic Systems," *CoRR*, p. 16, 2018. [Online]. Available: www.eecs.berkeley.edu/

[18] P. Geibel and F. Wysotzki, "Risk-sensitive Reinforcement Learning Applied to Control Under Constraints," *J. Artif. Int. Res.*, vol. 24, no. 1, pp. 81–108, jul 2005. [Online]. Available: http://dl.acm.org/citation.cfm?id=1622519.1622522 https://arxiv.org/pdf/1109.2147.pdf

[19] M. Turchetta, F. Berkenkamp, and A. Krause, "Safe Exploration in Finite Markov Decision Processes with Gaussian Processes," in *Advances in Neural Information Processing Systems*, 2016, pp. 4312–4320. [Online]. Available: https://arxiv.org/pdf/1606.04753.pdf

[20] S. Chernova and M. Veloso, "Confidence-based Policy Learning from Demonstration Using Gaussian Mixture Models," in *Proceedings of the 6th International Joint Conference on Autonomous Agents and Multiagent Systems*, ser. AAMAS '07. New York, NY, USA: ACM, 2007, pp. 233:1—-233:8. [Online]. Available: http://www.cs.cmu.edu/ mmv/papers/07aamas-sonia.pdf http://doi.acm.org/10.1145/1329125.1329407

[21] ——, "Multi-thresholded Approach to Demonstration Selection for Interactive Robot Learning," in *Proceedings of the 3rd ACM/IEEE International Conference on Human Robot Interaction*, ser. HRI '08. New York, NY, USA: ACM, 2008, pp. 225–232. [Online]. Available: http://doi.acm.org/10.1145/1349822.1349852 http://web.cs.wpi.edu/ rail/files/ChernovaVelosoHRI08.pdf

[22] B. Kim and J. Pineau, "Maximum Mean Discrepancy Imitation Learning." [Online]. Available: http://people.csail.mit.edu/beomjoon/publications/kim-rss13.pdf

[23] J. Zhang and K. Cho, "Query-Efficient Imitation Learning for End-to-End Simulated Driving," in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)*, 2017, pp. 2891–2897. [Online]. Available: https://arxiv.org/pdf/1605.06450.pdf

[24] K. Menda, K. Driggs-Campbell, and M. J. Kochenderfer, "DropoutDAgger: A Bayesian Approach to Safe Imitation Learning," 2017. [Online]. Available: https://arxiv.org/pdf/1709.06166.pdf

[25] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning," in *International Conference on Machine Learning (ICML)*, 2016. [Online]. Available: https://arxiv.org/pdf/1506.02142.pdf http://arxiv.org/abs/1506.02142

[26] S. Chernova, "Confidence-Based Robot Policy Learning from Demonstration," Ph.D. dissertation, Carnegie Mellon University, 2009. [Online]. Available: http://www.cs.cmu.edu/ mmv/papers/ChernovaThesis.pdf

[27] S. Zhang, Y. Jiang, S. Guni, and P. Stone, "Multirobot Symbolic Planning under Temporal Uncertainty." [Online]. Available: www.ifaamas.org

[28] M. Diaz, "Interactive and uncertainty-aware imitation learning: Theory and applications," Master's thesis, Concordia University, 2018.

[29] Y. C. B. M. L. P. Maxime Chevalier-Boisvert Florian Golemo, "Duckietown Environments for OpenAI Gym," \url{https://github.com/duckietown/gym-duckietown}, 2018.

[30] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "OpenAI Gym," 2016.

[31] H. He, H. Daumé Iii, J. Eisner, H. Daume, and H. Daumé, "Imitation Learning by Coaching," *Advances in Neural Information Processing Systems*, no. Section 5, pp. 3158–3166, 2012. [Online]. Available: https://papers.nips.cc/paper/4545-imitation-learning-by-coaching.pdf http://machinelearning.wustl.edu/mlpapers/papers/NIPS2012_1449

[32] J. Duchi, J. B. Edu, E. Hazan, and Y. Singer, "Adaptive Subgradient Methods for Online Learning and Stochastic Optimization *," *Journal of Machine Learning Research*, vol. 12, pp. 2121–2159, 2011. [Online]. Available: http://www.jmlr.org/papers/volume12/duchi11a/duchi11a.pdf