Verifying Robustness of Human-Aware Autonomous Cars

Dorsa Sadigh * **S. Shankar Sastry** ** **Sanjit A. Seshia** **

* *Stanford University, (e-mail: dorsa@cs.stanford.edu).*
** *UC Berkeley, (e-mail: {sseshia, sastry}@eecs.berkeley.edu)*

Abstract:
As human-robot systems make their ways into our every day life, safety has become a core concern of the learning algorithms used by such systems. Examples include semi-autonomous vehicles such as automobiles and aircrafts. The robustness of controllers in such systems relies on the accuracy of models of human behavior. In this paper, we propose a systematic methodology for analyzing the robustness of learning-based control of human-cyber-physical systems. We focus on the setting where human models are *learned* from data, with humans modeled as approximately rational agents optimizing their reward functions. In this setting, we provide a novel optimization-driven approach to find small deviations in learned human behavior that lead to violation of desired (safety) objectives. Our approach is experimentally validated via simulation for the application of autonomous driving.

Keywords: Urban mobility, Shared control, Human-Cyber-Physical Systems, Human-Robot Interaction, Autonomous Driving, Formal Verification

1. INTRODUCTION

Many cyber-physical systems and robots today operate with human input or interact closely with humans. Examples include fly-by-wire aircraft (interacting with pilots or ground-based operators), semi-autonomous automobiles (interacting with a driver, pedestrians, or other human-driven vehicles), and medical devices (interacting with a doctor, nurse, or patient). We term such system as *human-cyber-physical systems* or *human-robot systems*. Several of these systems are safety-critical, with severe costs of incorrect operation. Moreover, the interaction between human operators and autonomous controllers are often the reason for failures or "near failures", as noted by several studies (e.g., [FAA], L. T. Kohn and J. M. Corrigan and M. S. Donaldson, editors. [2000]).

In order to design such systems, it is crucial to have good models of human behavior. But how does one formally model human behavior? Broadly speaking, modeling approaches fall into two categories. On the one hand, one can rely on an *expert* to create a human model from experience, data, and documentation (e.g., Rushby [2002]). In such approaches, uncertainty in human behavior is modeled as adversarial, non-deterministic behavior, which can be extremely conservative. Additionally, when models are crafted by human experts, they are also prone to errors or biases by those experts. An alternative is to use a *data-driven approach*, where human models are inferred from data collected from experiments or the field. Such an approach has been gaining currency in recent times, notably for the application of autonomous driving Levine and Koltun [2012], Sadigh et al. [2014, 2016a,b], Shia et al. [2014], Vasudevan et al. [2012]. However, data-driven methods suffer the limitation that the learned models may be inaccurate due to lack of sufficient data or due to implicit assumptions in the learning process. Nonetheless, it is important to raise the level of assurance on the overall human-cyber-physical systems even when the learned model of human behavior may be inaccurate.

In this paper, we present a new approach for a rigorous analysis of human-cyber-physical systems that is based on *learned* models of human behavior. We focus on analyzing the robustness of a learning-based controller for human-cyber-physical systems where the human behavior deviates slightly from a learned model. As a motivating application, we consider autonomous controllers for vehicles that are based on the common and effective model predictive control (MPC) paradigm Bemporad and Morari [1999], Morari et al. [1993], in the receding horizon sense. In this setting, the autonomous controller computes its control actions for a horizon of N time units, while optimizing its reward function (which typically encodes safety and other objectives). For this purpose, it uses a dynamical model of both the system under control and its environment. Additionally, the environment contains human agents who are modeled as rational agents seeking to take actions that maximize their reward function. However, the human's *true* reward function is unknown to the autonomous controller. Therefore, the controller learns an approximation of the reward function from collected data, and uses this learned function in computing its actions. We refer to such a controller as an *interaction-aware controller*. Given a parameter δ that quantifies the deviation of the true human model from the learned model, our goal is to determine whether there exist behaviors of the human agent that can lead to a "failure", i.e., where the controller fails to meet its objectives. Since, given a large enough deviation, a failure can typically be induced, the utility of such an analysis is

also to determine the value of δ at which a failure can occur; hence, we also refer to the problem as one of *falsification*.

Verifying robustness of such human-cyber-physical systems faces some unique challenges. First, since the human model is learned from incomplete data under specific assumptions and biases of the learning algorithm, and the true reward function is unknown, the learned model will be inaccurate. In such a setting, how can one bound the error in the learned model? Second, given an error bound, how can one efficiently verify whether the controller will always achieve the desired objectives in an environment where the human agent(s) deviate from the learned model within that error bound?

Our approach addresses both challenges using optimization-based algorithmic methods. For the first, we provide a systematic and efficient methodology for estimating a bound between the values of the human's true reward function and the controller's estimate. Our methodology generates experiments to use in human subject trials during the learning process. For the second challenge, we present a novel encoding of the robustness verification problem from an optimization problem involving quantifiers over reward functions to a more straightforward quantifier-free optimization problem.

To the best of our knowledge, this paper is one of the first attempts to bring a systematic verification methodology to interaction-aware controllers for human-cyber-physical systems. More specifically, this paper makes the following novel contributions:

- A formalization of the problem of verifying robustness of controllers for human-cyber-physical systems with learned human models (Sec. 4);
- An encoding of the above problem into a more efficiently-solvable optimization problem (Sec. 4);
- An efficient (almost linear time) optimization-driven approach to estimating an error bound, δ, between the true human reward function and the controller's estimate (Sec. 5), and
- An experimental evaluation of our approach for autonomous vehicles using a car simulator (Sec. 6).

2. RUNNING EXAMPLE

We focus on falsification for human-cyber-physical systems, where a robot (e.g., an autonomous vehicle) interacts with human agents (e.g., human-driven vehicles) in the environment. Our goal is to verify if any of the actions of the human within a bound of a learned human model can possibly cause a violation of the desired specification, such as a collision between the two vehicles (autonomous and human-driven). For instance, consider an autonomous car (white car) driving on a road shared with a human-driven vehicle (orange car) (Fig. 1). The autonomous car would normally optimize for reaching its destination, which in this example is to safely change lanes. The white car would choose its actions based on considering its interaction with the human-driven vehicle, and using a learned model of the orange car. Here, the white car decides to start the lane change maneuver because it has learned a model of the human that suggests the orange car would also optimize for her own safety, and thus will follow trajectory A.

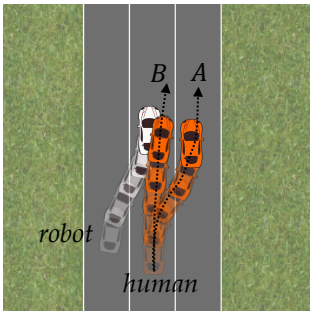

Fig. 1. Based on a learned model of the human, the autonomous car (white) decides to change lane assuming the human-driven car (orange) follows trajectory A. In practice the orange car can possibly follow a perturbed model resulting in trajectory B.

Similar to all learned models, this model of the human (orange car) is learned under a set of assumptions from a limited training dataset. So the truth is *the actual human can possibly take actions that slightly differ from the robot's learned human model*. For example, the orange car might decide to choose a slightly perturbed trajectory such as trajectory B. As it is clear in Fig. 1, such a perturbation can jeopardize satisfaction of the desired properties such as collision avoidance.

In this work, we show that these perturbations in models of learned human drivers exist, and we provide an efficient algorithm (almost linear) to learn a distribution for such possible perturbations using a query-based method. Further, we provide a method for finding the falsifying actions of the human agent (e.g. actions that result in trajectory B) within the perturbation bound.

3. PRELIMINARIES

In this section, we discuss some of the preliminaries required for understanding interaction-aware control of human-robot systems and learning human models through inverse reinforcement learning. In this work, we restrict ourselves to the case where a single human agent interacts with a single robot. We denote the human as $\mathcal{H}$ and the robot as $\mathcal{R}$. We broadly follow the approach of Sadigh et al. [2016b].

3.1 Interaction-Aware Control

We model the interaction between $\mathcal{H}$ and $\mathcal{R}$ as a fully observable dynamical system, similar to Sadigh et al. [2016b]. $x \in X$ denotes the continuous state of the system that lies in the set of all possible states X. For our running example, this will include the position, velocity, and heading of both agents $\mathcal{H}$ and $\mathcal{R}$ (orange and white cars). The robot can take continuous actions $u_{\mathcal{R}}$, which will modify the state of the system x through its dynamics. Similarly, the continuous actions of the human $u_{\mathcal{H}}$ affects the state of the system through the dynamics of $\mathcal{H}$. The dynamics of the two agents advances the state at every time step. For instance, these actions can be acceleration and steering angle for the vehicles shown in Fig. 1. Therefore, the overall dynamics is:

$$x^{t+1} = f(x^t, u^t_{\mathcal{R}}, u^t_{\mathcal{H}}). \quad (1)$$

The robot needs to calculate an autonomous control action $u_{\mathcal{R}}$ considering the interaction between the human and

robot. In this paper, our algorithm is oblivious to the source of $u_{\mathcal{R}}$; we give an approach based on nested-optimization for computing $u_{\mathcal{R}}$, but our falsification algorithm works for any other source of $u_{\mathcal{R}}$, e.g., deep neural networks, sampling based methods, or optimization-based techniques.

One approach in designing an autonomous controller for the robot is by optimizing its reward function. Assuming the actions of one agent affects the other, the reward function of $\mathcal{R}$ at every time step will depend on both $u_{\mathcal{R}}$ and $u_{\mathcal{H}}$. Therefore, this reward function at every step $r_{\mathcal{R}}(x^t, u^t_{\mathcal{R}}, u^t_{\mathcal{H}})$ requires a *model* describing the actions of the human $u^t_{\mathcal{H}}$. For instance, the white car's reward function in Fig. 1 is changing lanes while avoiding collisions with the orange car and following the rules of the road. The white car decides on its actions based on a model of how the orange car's driver would act on the road.

To optimize for the actions of the robot, we use a *model predictive control* (MPC) Morari et al. [1993] approach, where the robot plans for its actions by optimizing the reward for a finite horizon, and after taking the optimal action for the first time step, it will replan for the same horizon. Let $\mathbf{x} = (x^1, \ldots, x^N)^\top$ be a finite sequence of states, $\mathbf{u}_{\mathcal{R}} = (u^1_{\mathcal{R}}, \ldots, u^N_{\mathcal{R}})^\top$ a finite sequence of continuous actions of $\mathcal{R}$, and $\mathbf{u}_{\mathcal{H}} = (u^1_{\mathcal{H}}, \ldots, u^N_{\mathcal{H}})$ a finite sequence of continuous actions of $\mathcal{H}$. We let $R_{\mathcal{R}}$ be the accumulated reward of the robot for this finite horizon: $R_{\mathcal{R}}(x^0, \mathbf{u}_{\mathcal{R}}, \mathbf{u}_{\mathcal{H}}) = \sum_{t=1}^{N} r_{\mathcal{R}}(x^t, u^t_{\mathcal{R}}, u^t_{\mathcal{H}})$. Here, x^0 is the initial state of the horizon, and it will be updated based on the dynamics in eq. (1). We formally define the control problem for the robot as:

Problem 1. (Interaction-Aware Planner). Given a system as in eq. (1), an initial state $x^0 \in X$, a finite horizon N, and a robot reward function $R_{\mathcal{R}}$, compute:

$$\mathbf{u}^*_{\mathcal{R}} = \arg\max_{\mathbf{u}_{\mathcal{R}}} R_{\mathcal{R}}(x^0, \mathbf{u}_{\mathcal{R}}, \mathbf{u}^*_{\mathcal{H}}(x^0, \mathbf{u}_{\mathcal{R}})) \tag{2}$$

Remark 1. We emphasize again that Problem 1 is only one method for computing the actions of the robot that we have introduced for concreteness. Our approach may be combined with any other technique for computing control actions over a finite horizon.

The reward function $R_{\mathcal{R}}$ takes into account the interaction between the two agents by depending on the actions of the human $\mathbf{u}^*_{\mathcal{H}}(x^0, \mathbf{u}_{\mathcal{R}})$. These actions can be computed based on a data-driven model of $\mathcal{H}$. Ideally, humans are modeled as agents who optimize their reward function $R_{\mathcal{H}}$ as well:

$$\mathbf{u}^*_{\mathcal{H}}(x^0, \mathbf{u}_{\mathcal{R}}) = \arg\max_{\mathbf{u}_{\mathcal{H}}} R_{\mathcal{H}}(x^0, \mathbf{u}_{\mathcal{R}}, \mathbf{u}_{\mathcal{H}}) \tag{3}$$

Such a reward function represents our data-driven model of $\mathcal{H}$, which is learned offline. We let $R_{\mathcal{H}}(x^0, \mathbf{u}_{\mathcal{R}}, \mathbf{u}_{\mathcal{H}})$ be the sum of reward functions of $\mathcal{H}$ over horizon N, where $r_{\mathcal{H}}(x^t, u^t_{\mathcal{R}}, u^t_{\mathcal{H}})$ is the human's reward function at every time step: $R_{\mathcal{H}}(x^0, \mathbf{u}_{\mathcal{R}}, \mathbf{u}_{\mathcal{H}}) = \sum_{t=1}^{N} r_{\mathcal{H}}(x^t, u^t_{\mathcal{R}}, u^t_{\mathcal{H}})$. For our running example, the reward function of the human (orange car) can include collision avoidance and following the rules of the road.

3.2 Inverse Reinforcement Learning (IRL)

The human model we have used in this work is based on learning $R_{\mathcal{H}}$ from demonstrated trajectories provided by $\mathcal{H}$ Abbeel and Ng [2004]. However, we note that our falsification algorithm is oblivious to the specific learning method used for finding such a human model. We parameterize the human's reward function at every time step as a linear combination of a set of hand-coded features: $r_{\mathcal{H}}(x^t, u^t_{\mathcal{R}}, u^t_{\mathcal{H}}) = w \cdot \phi(x^t, u^t_{\mathcal{R}}, u^t_{\mathcal{H}})$. Here, $\phi(x^t, u^t_{\mathcal{R}}, u^t_{\mathcal{H}})$ is a vector of the features, and w is a vector of the weights corresponding to each feature. The features describe different aspects of the environment or the robot that the human should care about. For instance, for our running example, we choose 5 features that are based on: i) distance to the boundaries of the road; ii) distance to the middle of the lane; iii) heading; iv) velocity, and v) distance to the other vehicles on the road for collision avoidance. We use Gaussian kernels for the features that encode distances.

We then apply the maximum entropy principle Levine and Koltun [2012], Ziebart [2010], Ziebart et al. [2008] in order to recover the weights w from demonstrated trajectories. Under this principle, the actions of $\mathcal{H}$ are taken probabilistically, where higher reward $r_{\mathcal{H}}$ corresponds to more probable trajectories: $P(\mathbf{u}_{\mathcal{H}}|x^0, w) \propto \exp(R_{\mathcal{H}}(x^0, \mathbf{u}_{\mathcal{R}}, \mathbf{u}_{\mathcal{H}}))$. The weights can be recovered from the demonstrated trajectories by finding the maximum likelihood: $\max_w P(\mathbf{u}_{\mathcal{H}}|x^0, w)$. The reward function $r_{\mathcal{H}}$ is then reconstructed from the estimated weights.

4. ROBUSTNESS IN H-CPS CONTROL

Our falsification problem is to find out whether there exists a sequence of human actions that can lead to unsafe scenarios. Let $R_{\mathcal{H}}$, i.e., the sum of the learned reward functions of $\mathcal{H}$ over horizon N represent our human model. Our ultimate goal is to verify that the robot is resilient to model inaccuracies of $R_{\mathcal{H}}$ within a specific bound. Such inaccuracies usually exist due to two main factors: i) particular assumptions on the learning algorithm, ii) insufficiency of collected training data.

We have assumed humans are optimizers of a particular type of reward function $R_{\mathcal{H}}$, and such reward functions can be learned through various learning techniques (e.g., IRL as in Sec. 3.2). However, in practice the human might follow a different reward function, which we call the true reward function $R^\dagger_{\mathcal{H}}$. $R^\dagger_{\mathcal{H}}$ can possibly have a different structure from $R_{\mathcal{H}}$, i.e., it might not even be a linear combination of a set of hand-coded features described in Sec. 3.2, or we (designers) might have missed specifying a particular feature as part of ϕ. So the learning methods can possibly never converge to the true reward function $R^\dagger_{\mathcal{H}}$. Further, we might decide to learn $R_{\mathcal{H}}$ from a collection of human demonstrations, but in practice, the robot interacts with a specific human agent whose true reward function is $R^\dagger_{\mathcal{H}}$, which can be different from $R_{\mathcal{H}}$ due to variations amongst humans. So when it comes to interacting with different humans, the learned $R_{\mathcal{H}}$ might not perform as expected.

We are motivated to verify if Problem 1 can be solved in scenarios where the true human reward function $R^\dagger_{\mathcal{H}}$, unknown to us, deviates a bit from the learned function $R_{\mathcal{H}}$. We let δ to be the bound between the distance of these two reward functions:

$$\forall (x^0, \mathbf{u}_{\mathcal{R}}, \mathbf{u}_{\mathcal{H}}),\ |R_{\mathcal{H}}(x^0, \mathbf{u}_{\mathcal{R}}, \mathbf{u}_{\mathcal{H}}) - R^\dagger_{\mathcal{H}}(x^0, \mathbf{u}_{\mathcal{R}}, \mathbf{u}_{\mathcal{H}})| < \delta. \tag{4}$$

For brevity, we will slightly abuse notation and write this inequality as $|R_{\mathcal{H}} - R^\dagger_{\mathcal{H}}| < \delta$.

Problem 2. (Falsification). For a parameter δ describing the bound between the human's true and learned reward functions ($R^\dagger_\mathcal{H}$ and $R_\mathcal{H}$), compute a falsifying input $\tilde{\mathbf{u}_\mathcal{H}}$:

$$\begin{aligned}\tilde{\mathbf{u}_\mathcal{H}} = \arg\min_{\mathbf{u}_\mathcal{H}} \quad & R_\mathcal{R}(x^0, \mathbf{u}^*_\mathcal{R}, \mathbf{u}_\mathcal{H}) \\ \text{subject to} \quad & \exists R^\dagger_\mathcal{H} : \quad \mathbf{u}_\mathcal{H} = \arg\max_{\bar{\mathbf{u}_\mathcal{H}}} R^\dagger_\mathcal{H}(x^0, \mathbf{u}^*_\mathcal{R}, \bar{\mathbf{u}_\mathcal{H}}), \\ & |R_\mathcal{H} - R^\dagger_\mathcal{H}| < \delta.\end{aligned} \tag{5}$$

Here, $\mathbf{u}^*_\mathcal{R}$ is computed by solving the nested optimization in Problem 1, where $\mathbf{u}^*_\mathcal{H}(x^0, \mathbf{u}_\mathcal{R})$ is the true optimizer of the learned reward function $R_\mathcal{H}$.

Problem 2 looks for the worst case input provided by the human that violates the specification by minimizing the reward function of the robot $R_\mathcal{R}$, while still remaining in the set of possible human models, i.e., this input can optimize an alternative reward $R^\dagger_\mathcal{H}$ that lies within δ distance of the learned reward function $R_\mathcal{H}$. We now show how we optimally solve for this falsifying input.

Theorem 1. The optimization in Problem 2 is equivalent to solving the following:

$$\begin{aligned}\tilde{\mathbf{u}_\mathcal{H}} = \arg\min_{\mathbf{u}_\mathcal{H}} \quad & R_\mathcal{R}(x^0, \mathbf{u}^*_\mathcal{R}, \mathbf{u}_\mathcal{H}) \\ \text{subject to} \quad & R_\mathcal{H}(x^0, \mathbf{u}^*_\mathcal{R}, \mathbf{u}_\mathcal{H}) > R^*_\mathcal{H} - 2\delta,\end{aligned} \tag{6}$$

where $R^*_\mathcal{H} = R_\mathcal{H}(x^0, \mathbf{u}^*_\mathcal{R}, \mathbf{u}^*_\mathcal{H}(x^0, \mathbf{u}^*_\mathcal{R}))$ is a constant evaluated at $\mathbf{u}^*_\mathcal{R}$ and $\mathbf{u}^*_\mathcal{H}(x^0, \mathbf{u}^*_\mathcal{R})$. Here, $\mathbf{u}^*_\mathcal{R}$ is computed by solving Problem 1, which depends on the function representing the optimal actions of the human $\mathbf{u}^*_\mathcal{H}(x^0, \mathbf{u}_\mathcal{R})$. Using this $\mathbf{u}^*_\mathcal{R}$, we then find $\mathbf{u}^*_\mathcal{H}(x^0, \mathbf{u}^*_\mathcal{R}) = \arg\max_{\mathbf{u}_\mathcal{H}} R_\mathcal{H}(x^0, \mathbf{u}^*_\mathcal{R}, \mathbf{u}_\mathcal{H})$.

Proof 1. To show the equivalence between equation (5) and (6), we need to consider the equivalence of the constraints.

First, we show $\exists R^\dagger_\mathcal{H} : \mathbf{u}_\mathcal{H} = \arg\max_{\bar{\mathbf{u}_\mathcal{H}}} R^\dagger_\mathcal{H}(x^0, \mathbf{u}^*_\mathcal{R}, \bar{\mathbf{u}_\mathcal{H}})$ will imply the constraint in equation (6).

Since $\exists R^\dagger_\mathcal{H} : \mathbf{u}_\mathcal{H} = \arg\max_{\bar{\mathbf{u}_\mathcal{H}}} R^\dagger_\mathcal{H}(x^0, \mathbf{u}^*_\mathcal{R}, \bar{\mathbf{u}_\mathcal{H}})$, we conclude that $R^\dagger_\mathcal{H}$ evaluated at the optimum $\mathbf{u}_\mathcal{H}$ is greater than $R^\dagger_\mathcal{H}$ evaluated at any other sequence of human actions such as $\mathbf{u}^*_\mathcal{H}(x^0, \mathbf{u}^*_\mathcal{R})$ (to simplify the notation, we simply let $\mathbf{u}^*_\mathcal{H} = \mathbf{u}^*_\mathcal{H}(x^0, \mathbf{u}^*_\mathcal{R})$):

$$R^\dagger_\mathcal{H}(x^0, \mathbf{u}^*_\mathcal{R}, \mathbf{u}_\mathcal{H}) \geq R^\dagger_\mathcal{H}(x^0, \mathbf{u}^*_\mathcal{R}, \mathbf{u}^*_\mathcal{H}). \tag{7}$$

Also, because of the existing bound δ, we can infer the following:

$$R_\mathcal{H}(x^0, \mathbf{u}^*_\mathcal{R}, \mathbf{u}_\mathcal{H}) > R^\dagger_\mathcal{H}(x^0, \mathbf{u}^*_\mathcal{R}, \mathbf{u}_\mathcal{H}) - \delta, \tag{8}$$

$$R^\dagger_\mathcal{H}(x^0, \mathbf{u}^*_\mathcal{R}, \mathbf{u}^*_\mathcal{H}) > R_\mathcal{H}(x^0, \mathbf{u}^*_\mathcal{R}, \mathbf{u}^*_\mathcal{H}) - \delta. \tag{9}$$

Equations (7) and (8) will result in:

$$R_\mathcal{H}(x^0, \mathbf{u}^*_\mathcal{R}, \mathbf{u}_\mathcal{H}) > R^\dagger_\mathcal{H}(x^0, \mathbf{u}^*_\mathcal{R}, \mathbf{u}^*_\mathcal{H}) - \delta. \tag{10}$$

Finally, equations (9) and (10) provide:

$$R_\mathcal{H}(x^0, \mathbf{u}^*_\mathcal{R}, \mathbf{u}_\mathcal{H}) > R_\mathcal{H}(x^0, \mathbf{u}^*_\mathcal{R}, \mathbf{u}^*_\mathcal{H}) - 2\delta, \tag{11}$$

which is the same as the constraint in equation (6):

$$R_\mathcal{H}(x^0, \mathbf{u}^*_\mathcal{R}, \mathbf{u}_\mathcal{H}) > R^*_\mathcal{H} - 2\delta. \tag{12}$$

Now, to complete the proof, we need to show that this constraint will result in the constraint in equation (5), i.e., $\exists R^\dagger_\mathcal{H} : \mathbf{u}_\mathcal{H} = \arg\max_{\bar{\mathbf{u}_\mathcal{H}}} R^\dagger_\mathcal{H}(x^0, \mathbf{u}^*_\mathcal{R}, \bar{\mathbf{u}_\mathcal{H}})$. Our approach is to construct the following reward function $R^\dagger_\mathcal{H}$ so that

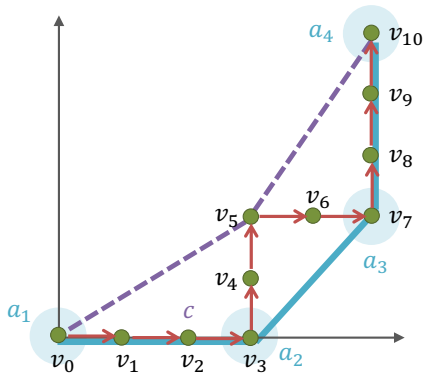

Fig. 2. Visualization tool representing 10 queries from the human. We move right for positive queries ($s_i = +1$), and move up for negative ones ($s_i = -1$) over $\mathbb{Z}^2$. The convex hull of this plot provides our maximum likelihood estimate of the error distribution.

at the optimal $\mathbf{u}_\mathcal{H}$ (i.e. $\mathbf{u}_\mathcal{H} = \arg\max_{\bar{\mathbf{u}_\mathcal{H}}} R^\dagger_\mathcal{H}(x^0, \mathbf{u}^*_\mathcal{R}, \bar{\mathbf{u}_\mathcal{H}})$), we will have:

$$R^\dagger_\mathcal{H}(x^0, \mathbf{u}^*_\mathcal{R}, \mathbf{u}_\mathcal{H}) = R_\mathcal{H}(x^0, \mathbf{u}^*_\mathcal{R}, \mathbf{u}_\mathcal{H}) + \delta, \tag{13}$$

and for any other human actions such as $\mathbf{u}'_\mathcal{H} \neq \mathbf{u}_\mathcal{H}$, this function is equal to:

$$R^\dagger_\mathcal{H}(x^0, \mathbf{u}^*_\mathcal{R}, \mathbf{u}'_\mathcal{H}) = R_\mathcal{H}(x^0, \mathbf{u}^*_\mathcal{R}, \mathbf{u}'_\mathcal{H}) - \delta. \tag{14}$$

Then, reordering the inequality in equation (5), and using equation (13) will result in:

$$\begin{aligned} & R_\mathcal{H}(x^0, \mathbf{u}^*_\mathcal{R}, \mathbf{u}_\mathcal{H}) + \delta > R^*_\mathcal{H} - \delta \\ \implies & R^\dagger_\mathcal{H}(x^0, \mathbf{u}^*_\mathcal{R}, \mathbf{u}_\mathcal{H}) > R^*_\mathcal{H} - \delta \end{aligned} \tag{15}$$

Also, by optimality of $R^*_\mathcal{H}$ and equation (14), we know for any $\mathbf{u}'_\mathcal{H} \neq \mathbf{u}_\mathcal{H}$:

$$R^\dagger_\mathcal{H}(x^0, \mathbf{u}^*_\mathcal{R}, \mathbf{u}'_\mathcal{H}) = R_\mathcal{H}(x^0, \mathbf{u}^*_\mathcal{R}, \mathbf{u}'_\mathcal{H}) - \delta \leq R^*_\mathcal{H} - \delta \tag{16}$$

Using the two results in equations (15) and (16), we conclude:

$$\begin{aligned} & R^\dagger_\mathcal{H}(x^0, \mathbf{u}^*_\mathcal{R}, \mathbf{u}_\mathcal{H}) \geq R^\dagger_\mathcal{H}(x^0, \mathbf{u}^*_\mathcal{R}, \mathbf{u}'_\mathcal{H}) \\ \implies & \mathbf{u}_\mathcal{H} = \arg\max_{\bar{\mathbf{u}_\mathcal{H}}} R^\dagger_\mathcal{H}(x^0, \mathbf{u}^*_\mathcal{R}, \bar{\mathbf{u}_\mathcal{H}}). \end{aligned} \tag{17}$$

So for the $R^\dagger_\mathcal{H}$ we have constructed, the constraint in equation (5) holds. Since the two constraints in (5) and (6) are equivalent, Theorem 1 holds true.

Through Theorem 1 we have shown we can solve eq. (6), a much simpler optimization problem, instead of solving Problem 2 to find the falsifying actions of $\mathcal{H}$ for a fixed given bound δ.

5. LEARNING THE ERROR BOUND

In this section, we provide an efficient algorithm for computing a reasonable error bound δ, as defined in eq. (4), which is used in solving Problem 2. We will first learn a distribution for the error:

$$\Delta(\xi) = R_\mathcal{H}(\xi) - R^\dagger_\mathcal{H}(\xi), \tag{18}$$

and then we upper bound $\Delta(\xi)$ by a constant factor δ using the distribution's quantiles. In the rest of the paper, we use Δ instead of $\Delta(\xi)$ for brevity. So assuming that $\Delta \sim \mathcal{D}$ is a random variable drawn from a distribution $\mathcal{D}$, our goal is to *efficiently* learn $\mathcal{D}$. We propose using a query based method that efficiently (almost linear time in the number

of queries, i.e., $O(n \log n)$) finds the maximum likelihood estimate for $\mathcal{D}$.

Suppose, we query the human $\mathcal{H}$ whether she prefers trajectory ξ_A or ξ_B. Then, if the user decides ξ_A is preferred over ξ_B, we can conclude that the true reward function of trajectory ξ_A is higher than the true reward function of trajectory ξ_B, i.e.,

$$R^\dagger_{\mathcal{H}}(\xi_A) > R^\dagger_{\mathcal{H}}(\xi_B). \tag{19}$$

Note, since the trajectory is fully determined by $(x^0, \mathbf{u}_{\mathcal{H}}, \mathbf{u}_{\mathcal{R}})$, we alternatively can write $R_{\mathcal{H}}$ as a function of ξ. Using the error bound we have

$$R_{\mathcal{H}}(\xi_A) + \Delta_A > R_{\mathcal{H}}(\xi_B) + \Delta_B. \tag{20}$$

We also assume $\Delta_A, \Delta_B \sim \mathcal{D}$ are independent random variables, and so is their difference: $\chi = \Delta_A - \Delta_B$. We can rewrite this as

$$\chi = \Delta_A - \Delta_B > R_{\mathcal{H}}(\xi_B) - R_{\mathcal{H}}(\xi_A), \tag{21}$$

where $R_{\mathcal{H}}$ is the learned reward function. Here, we simply let $R = R_{\mathcal{H}}(\xi_B) - R_{\mathcal{H}}(\xi_A)$.

Therefore, assuming humans are consistent and always respond with a preference, every query from the human for a fixed set of two trajectories ξ_A and ξ_B will results in either $\chi > R$ or $\chi < R$. Our goal in this section is to find the maximum likelihood distribution $\bar{\mathcal{D}}$ of χ by asking queries from the human for a number of times, and recovering the distribution of the true bound $\Delta \sim \mathcal{D}$.

Remark 2. Here the human responses to the queries only provide an estimate of the human's true reward function. These comparisons allow us to learn the human's preferences, which can possibly be different from the true reward function she would optimize in practice Basu et al. [2017], Sadigh et al. [2017]. However, for the purpose of falsification in this paper, the preference reward function is a reasonable estimate of $R^\dagger_{\mathcal{H}}$.

5.1 Learning the Error Distribution

Now, we propose our algorithm to find the distribution $\bar{\mathcal{D}}$ (where χ is drawn from this distribution: $\chi \sim \bar{\mathcal{D}}$) based on any M given comparison queries $\chi_i > R_i$ or $\chi_i < R_i$.

Suppose that $\chi_1, \dots, \chi_M$ are i.i.d. and drawn from some unknown distribution $\bar{\mathcal{D}}$. Given $R_1, \dots, R_M \in \mathbb{R}$, and the corresponding signs $s_1, \dots, s_M \in \{-1, +1\}$, representing the observations: $s_i(\chi_i - R_i) > 0$, i.e., $\chi_i > R_i$ for $s_i = +1$, and $\chi_i < R_i$ for $s_i = -1$, the goal is to find the maximum likelihood estimate of the distribution $\bar{\mathcal{D}}$.

Without loss of generality, we assume that R_i's are sorted, i.e. $R_1 < R_2 < \cdots < R_M$. Now let us define

$$p_i = \mathbb{P}_{\chi \sim \bar{\mathcal{D}}}[\chi < R_i]. \tag{22}$$

Since R_i's are sorted, the defined p_i's would be sorted:

$$p_1 \le p_2 \le \cdots \le p_M \tag{23}$$

The likelihood of our observations can be expressed in terms of p_i's. Note that any increasing sequence of p_i's that lie in $[0, 1]$ is valid. Our problem of estimating the distribution reduces to finding the increasing sequence of $p_1 \le \cdots \le p_M \in [0, 1]$ that maximizes the log-likelihood of our observations. We now propose our method that finds p_i's by solving the following constrained optimization:

$$\begin{aligned} \max_{p_1,\dots,p_M} \quad & \log\Big(\prod_{i:s_i=+1}(1-p_i) \prod_{i:s_i=-1} p_i\Big) \\ \text{subject to} \quad & 0 \le p_1 \le p_2 \le \cdots \le p_M \le 1. \end{aligned} \tag{24}$$

Suppose $p^*_1 \le \cdots \le p^*_M$ are the optimizers of eq. (24). We derive some necessary conditions that $p^*_1, \dots, p^*_M$ need to satisfy, and conclude that these uniquely determine p^*_i.

We can partition p^*_i into contiguous maximal subsets of equal values. For example, $p^*_a = p^*_{a+1} = \cdots = p^*_b$ represents such a maximal contiguous subset, i.e., $p^*_{a-1} < p^*_a$ and $p^*_b < p^*_{b+1}$. We let q represent the common value of this contiguous subset.

Lemma 1. In the above context, the value of q must be equal to: $\frac{n_-^{[a,b]}}{n_-^{[a,b]}+n_+^{[a,b]}}$, where $n_+^{[a,b]} = |\{i : s_i = +1 \wedge i \in [a, b]\}|$ and $n_-^{[a,b]} = |\{i : s_i = -1 \wedge i \in [a, b]\}|$.

Proof 2. If we perturb q by a small amount in any direction, we would still be satisfying the constraint of monotonicity of p_i's. Further, the derivative of the log-likelihood with respect to q must be 0 at the optimal values $q = p^*_a$. This means that:

$$-\sum_{i:s_i=+1 \wedge i \in [a,b]} \frac{1}{1-q} + \sum_{i:s_i=-1 \wedge i \in [a,b]} \frac{1}{q} = 0, \tag{25}$$

which results in $q = n_-^{[a,b]}/(n_-^{[a,b]} + n_+^{[a,b]})$.

We visualize the values $n_+^{[a,b]}$ and $n_-^{[a,b]}$ using the graph in Fig. 2: consider a path drawn on $\mathbb{Z}^2$, and let the path start from the origin $v_0 = (0, 0)$. For each i in the sorted order of χ_i, if $s_i = -1$, move up one unit, and if $s_i = +1$, move to the right one unit, and call the new point v_i. Note, at the beginning runs, we would move to the right more frequently, as the sorted data makes $s_i = +1$ results more likely. Fig. 2 shows an example of such a run.

Between every two points v_{a-1} and v_b on the plot the difference in the y-direction is $\Delta y = n_-^{[a,b]}$, and similarly the difference in the x-direction is $\Delta x = n_+^{[a,b]}$. Therefore, the optimal value of the log-likelihood for each maximal contiguous subset, e.g., $[a, b]$ is $q = \frac{\Delta y}{\Delta x + \Delta y}$, where $(\Delta x, \Delta y) = v_b - v_{a-1}$. We can think of $\frac{\Delta y}{\Delta x + \Delta y}$ as a type of slope, which we call L-slope. Note, the L-slope is an increasing function of the real slope (i.e., $\frac{\Delta y}{\Delta x}$). While the real slope lies in $[0, \infty]$, the L-slope has a monotonic relation to the real slope and maps it to $\frac{\Delta y}{\Delta y + \Delta x} \in [0, 1]$.

So far, we have shown that if we know the maximal contiguous partitions of p_i's, the optimal values p^*_i's are uniquely determined by the L-slopes. Thus, the main question is how to find such maximal contiguous segments.

First, let's assume there are k maximal contiguous segments, and we represent them with $a_1, \dots, a_k$, where $a_1 = v_0$, and $a_k = v_M$, and all the other a_j's for $j \in \{1, \dots, k\}$ are equal to vertices v_i that partition the p_i's into maximal contiguous pieces. We use l_i to denote the L-slope between each a_i and a_{i+1}. Note that because of ordering of $p_1 \le p_2 \le \cdots \le p_M$, these L-slopes are also in an increasing order $l_1 \le l_2 \le \cdots \le l_{k-1}$. Therefore, these L-slopes create a convex graph, i.e., a_j's lie in a convex position.

Lemma 2. In the above context, a_j's are precisely the set of vertices of the convex hull of the constructed graph.

Proof 3. We prove this by contradiction. Assume that a_j's are not the convex hull of the graph. For example in Fig. 2, this could happen with $a_1 = v_0, a_2 = v_5, a_3 = v_{10}$.

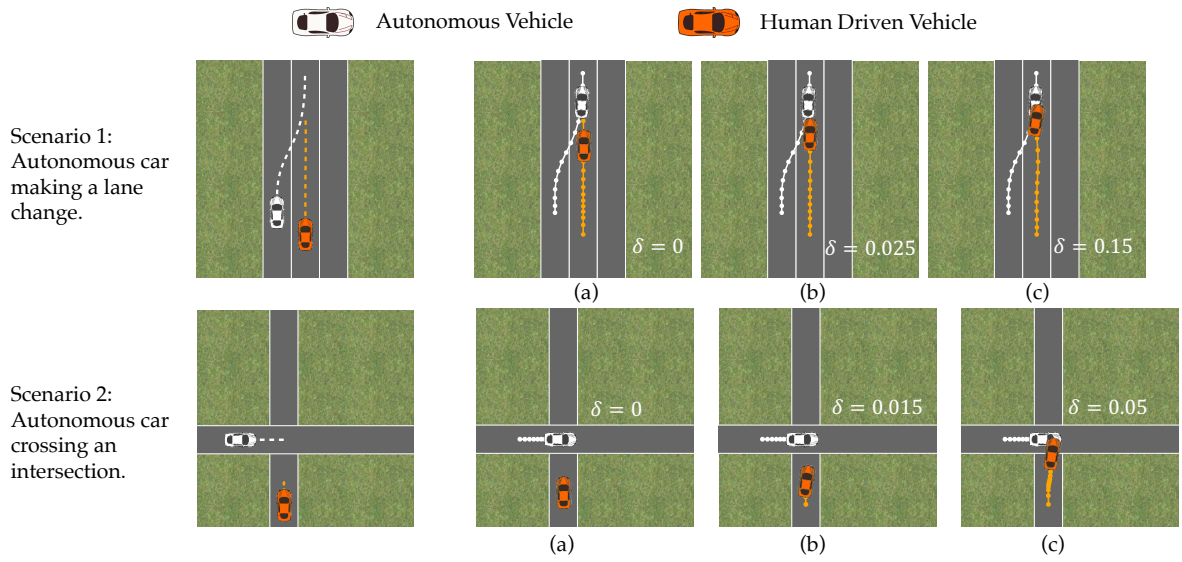

Fig. 3. Falsification in driving scenarios. Each row represents a scenario, where the cars start from the initial positions shown on the left, then (a) shows the optimal actions of human based on $R_{\mathcal{H}}$. In (b) and (c), we illustrate the optimal actions of the vehicles if $\mathcal{H}$ follows a perturbed reward function. The perturbation in (b) is not enough for violation of the safety property; however, in (c) with enough perturbation the two vehicles collide.

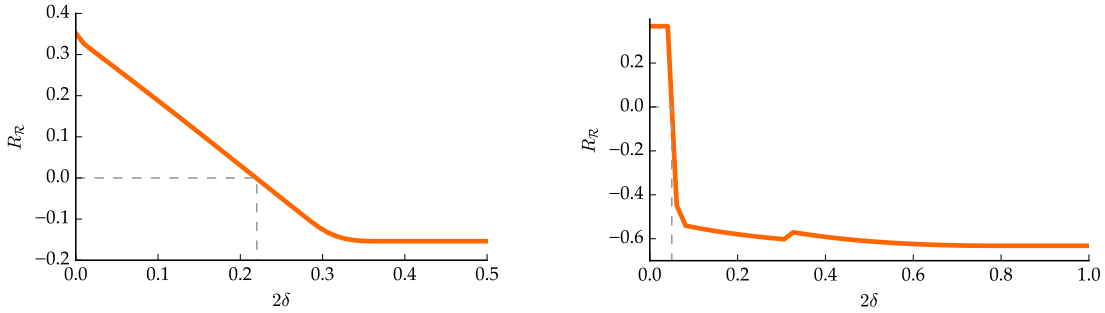

Fig. 4. Robot Reward function of scenario 1 shown in (a), and scenario 2 in (b) with respect to δ. The violation threshold in (a) is $\delta = 0.11$, and in (b) is $\delta = 0.025$.

Because a_j's are not the convex hull, there exists an index i and a point on the graph c such that c lies under the line segment between a_i and a_{i+1} (e.g., let $c = v_2$ in the segment connecting $a_1 = v_0$ to $a_2 = v_5$).

Consider the probabilities corresponding to the line segment from a_i to a_{i+1}. These would be $p_r, p_{r+1}, \ldots, p_s$, if $a_i = v_{r-1}$ and $a_{i+1} = v_s$; note that we have already shown all of them must be equal to the L-slope between a_i and a_{i+1}, i.e., $p_r = \cdots = p_s = l_i$. We partition these probabilities into two contiguous segments $p_r, \ldots, p_j$ and $p_{j+1}, \ldots, p_s$, where j is the index for which $c = v_j$.

We perturb the first segment, $p_r, \ldots, p_j$ by a small amount ϵ in the *negative* direction, and perturb the second segment, $p_{j+1}, \ldots, p_s$ by ϵ in the *positive* direction. Since we decrease the first segment and increase the second, this does not violate our ordering constraint $p_1 \leq \cdots \leq p_M$. Then, by a similar argument as in the proof of Lemma 1, we can conclude that this small perturbation increases the log-likelihood; in other words, the derivative of the log-likelihood with respect to the first segment is negative and the derivative with respect to the second segment is positive. We conclude that when a_j's form the optimal partition, no such point c can be found, and therefore the a_j's form the convex hull of the graph.

Theorem 2. The optimization problem in eq. (24) can be solved in $O(M \log(M))$ time.

Proof 4. Sorting the R_i's takes $O(M \log(M))$ time. As we have shown in Lemma 2, the optimal partition is determined by the convex hull of the constructed graph. The graph can be constructed in linear time, and the convex hull can be computed in linear time as well. Finally, the values p_i's are determined by the L-slopes, each of which can be computed in $O(1)$ time. Thus the bottleneck is in the initial sort, and the whole computation takes $O(M \log(M))$ time.

This method enables us to efficiently recover an estimate of the CDF of distribution $\bar{\mathcal{D}}$.

Remark 3. In practice, the CDF might be approximated well by a Gaussian. Finding the parameters of the Gaussian can again be formulated as a similar optimization in eq. (24), where the objective is still the log-likelihood of

the observed data, but the variables are replaced by the parameters of the Gaussian.

5.2 Recovering the Error Distribution

To recover the distribution of the error $\Delta \sim \mathcal{D}$, we need to make a further assumption that the distribution $\bar{\mathcal{D}}$ is symmetric. [1] Assuming we successfully find the distribution of $\chi \sim \bar{\mathcal{D}}$, we can use moment generating functions $M_\chi(t) = \mathbb{E}[e^{t\chi}]$ to recover $\mathcal{D}$. Given

$$\chi = \Delta_A - \Delta_B, \text{ where } \chi \sim \bar{\mathcal{D}} \text{ and } \Delta_A, \Delta_B \sim \mathcal{D} \quad (26)$$

$$\begin{aligned} M_\chi(t) &= M_{\Delta_A - \Delta_B}(t) = M_{\Delta_A}(t) \cdot M_{-\Delta_B}(t) \\ &= M_{\Delta_A}(t) \cdot M_{\Delta_B}(-t). \end{aligned} \quad (27)$$

Since Δ_A and Δ_B are drawn from the same distribution, their moment generating function is the same. Further, Δ is drawn from a symmetric distribution, i.e., $M_\Delta(t) = M_\Delta(-t)$, so: $M_\chi(t) = M_\Delta(t)^2$. Assuming we have computed $\chi \sim \bar{\mathcal{D}}$, we now can use this relation to find the distribution of $\Delta \sim \mathcal{D}$.

Remark 4. If we are only interested in solving eq. (6), we do not need to extract $\mathcal{D}$ from $\bar{\mathcal{D}}$. We can simply replace the 2δ bound by a number drawn from the quantiles of the distribution $\bar{\mathcal{D}}$. The reason for this is that even if we replace $|R^\dagger_{\mathcal{H}} - R_{\mathcal{H}}| \leq \delta$ by the weaker assumption:

$\forall \xi_A, \xi_B : |(R^\dagger_{\mathcal{H}}(\xi_A) - R^\dagger_{\mathcal{H}}(\xi_B)) - (R_{\mathcal{H}}(\xi_A) - R_{\mathcal{H}}(\xi_B))| \leq 2\delta$,

Theorem 1 would still hold (with essentially the same proof). Note that the above quantity is simply: $|\chi| = |\Delta_A - \Delta_B|$, whose distribution is given by $\bar{\mathcal{D}}$.

Remark 5. We select 2δ from the quantiles of $\bar{\mathcal{D}}$ in such a way that: $\Pr_{\chi \sim \bar{\mathcal{D}}}(|\chi| > 2\delta) < \epsilon$, where $\epsilon \in [0, 1]$ is the tolerated error ($1 - \epsilon$ is the confidence level). Our method enables us to provide statistical verification guarantees for the learned reward function.

6. CASE STUDIES

In this section, we demonstrate our falsification algorithm for two autonomous driving scenarios. We also efficiently learn an error distribution for human's reward function based on the proposed query-based method. We solved the constrained optimizations using the package Ipopt Wächter and Biegler [2006]. Our falsification algorithm is solved in less than a second (0.25s on average) when run on a 2.3 GHz Intel Core i7 processor.

Both vehicles follow a simple point-mass dynamics model. We let $\mathbf{x} = [x\ y\ \theta\ v]^\top$ denote the state of this dynamical system, where x and y are the position of the vehicle on the map, θ is its heading, and v is its velocity. Further, the control input $\mathbf{u} = [u_1\ u_2]^\top$ includes the steering input u_1, and acceleration u_2. Therefore, we define the dynamics of the vehicle (eq. (1)) to be:

$$\begin{aligned} \left[x^{t+1}\ y^{t+1}\ \theta^{t+1}\ v^{t+1}\right] = \left[x^t\ y^t\ \theta^t\ v^t\right] + \\ dt\left[v \cdot \cos(\theta)\ \ v \cdot \sin(\theta)\ \ v \cdot u_1\ \ u_2 - \alpha \cdot v\right]. \end{aligned}$$

Here, α is the friction coefficient. We illustrate the two case studies in Fig. 3. The top row corresponds to the autonomous car (white car) changing lane (the running example), and the bottom row corresponds to the autonomous car crossing an intersection. For both cases, the reward function includes collision avoidance between the vehicles, which we represent by a non-symmetric Gaussian centered at the other vehicle.

Scenario 1: Falsifying Collision Avoidance in a Lane Change. In this scenario, the autonomous car (white car in Fig. 3) optimizes for changing lanes, while avoiding collisions. Assuming that $\mathcal{H}$ fully follows the learned reward function $R_{\mathcal{H}}$, no collisions occur and the autonomous car safely changes lanes (Fig. 3(a)). However, the orange car might not exactly follow $R_{\mathcal{H}}$, so for a perturbation of $R_{\mathcal{H}}$ by $\delta = 0.025$, the two cars take a riskier maneuver (Fig. 3(b)), and finally for a sufficiently large $\delta = 0.15$, we find a falsifying set of actions for the human (Fig. 3(c)), which clearly ends in a collision.

We demonstrate $R_{\mathcal{R}}$ with respect to δ in Fig. 4(a) with respect to the reward, where $R_{\mathcal{R}} < 0$ corresponds to collisions. So the safety property of collision avoidance will be violated for any model of human $R^\dagger_{\mathcal{H}}$ that lies in a range that is at least $\delta = 0.11$ away from $R_{\mathcal{H}}$.

Scenario 2: Falsifying Collision Avoidance in an Intersection. In this scenario, our goal is to find the falsifying actions for the human-driven car (orange car) when the autonomous car (white car) plans to cross an intersection as shown in Fig. 3. Similar to the previous scenario, following $R_{\mathcal{H}}$, results in a collision-free outcome (Fig. 3(b)). However, with a slightly perturbed reward function of the human with $\delta = 0.015$, the orange car gets closer to the autonomous car, and for a slightly higher $\delta = 0.05$ the two cars collide as shown in Fig. 3(c).

Similar to the previous case, we have shown the reward function $R_{\mathcal{R}}$ with respect to δ in Fig. 4, which demonstrates the violation threshold is $\delta = 0.025$. We hypothesize that the bumps in the reward function (Fig. 4(b)) in this scenario are due to the system falling into local optima in solving the falsification problem.

Learning Human's Error. Using our query-based method, we learn a distribution for Δ, and find an appropriate bound δ. As shown in Fig. 5(a), we have done 100 queries from $\mathcal{H}$ asking the human to compare two provided trajectories. The blue line in Fig. 5(a) represents the walk over $\mathbb{Z}^2$ based on the sorted queries, where each horizontal line represents a positive query ($s_i = +1$), and each vertical line represents a negative query ($s_i = -1$) (similar to Fig. 2). The orange line is the convex hull of the graph facilitating estimation of the distribution of Δ.

In Fig. 5(b), we have found the maximum likelihood estimate of the CDF of the distribution from the convex hull in Fig. 5(a). Similarly, we can fit a Gaussian instead as shown in purple. Using the estimated CDF, we choose $\delta = 0.1$ for confidence level of $\epsilon = 0.74$ as discussed in Remark 5. Note, with this confidence level and bound $\delta = 0.1$, we would have violated the specification for the second case study (crossing an intersection), while still satisfying collision avoidance in the first scenario (lane change) based on the thresholds found in Fig. 4.

[1] Since we only observe the difference between two samples from this distribution, we can only have information about the even moments of the distribution. Hence, we have to make an assumption about the odd moments, i.e., they have to be zero, which is equivalent to assuming the distribution is symmetric.

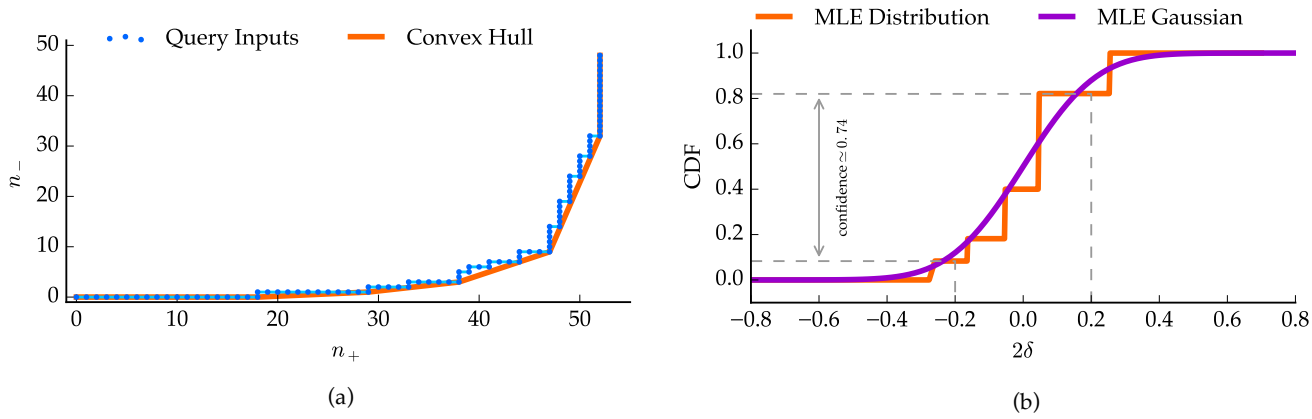

Fig. 5. Learning Human's Error Distribution. In (a) we visualize the comparison queries from $\mathcal{H}$, and find the convex hull of the graph allowing us to find the CDF of the error distribution as shown in orange in (b). Using different quantiles of the CDF, we set an appropriate bound for δ.

7. CONCLUSION

We have formalized the problem of falsification for human-cyber-physical systems and provided an optimization-based approach for solving it. Our autonomous driving case studies provide evidence of the utility of this approach. For future work, we plan to expand the application of this methodology to a broader class of human behavior models and human-cyber-physical systems. We note that our high-level approach might be more broadly applicable to algorithmically analyzing the behavior of other kinds of systems involving models learned from data.

8. ACKNOWLEDGEMENTS

Toyota Research Institute ("TRI") provided funds to assist the authors with their research but this article solely reflects the opinions and conclusions of its authors and not TRI or any other Toyota entity. We also would like to acknowledge the NSF VeHICaL project (CNS-1545126).

REFERENCES

Abbeel, P. and Ng, A.Y. (2004). Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, 1. ACM.

Basu, C., Yang, Q., Hungerman, D., Singhal, M., and Dragan, A.D. (2017). Do you want your autonomous car to drive like you? In *Proceedings of the 2017 ACM/IEEE International Conference on Human-Robot Interaction*, 417–425. ACM.

Bemporad, A. and Morari, M. (1999). Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35(3), 407–427. doi:10.1016/S0005-1098(98)00178-2. URL `http://dx.doi.org/10.1016/S0005-1098(98)00178-2`.

(FAA), F.A.A. (1995). The interfaces between flight crews and modern flight systems. `http://www.faa.gov/avr/afs/interfac.pdf`.

L. T. Kohn and J. M. Corrigan and M. S. Donaldson, editors. (2000). To err is human: Building a safer health system. Technical report, A report of the Committee on Quality of Health Care in America, Institute of Medicine, Washington, DC. National Academy Press.

Levine, S. and Koltun, V. (2012). Continuous inverse optimal control with locally optimal examples. *arXiv preprint arXiv:1206.4617*.

Morari, M., Garcia, C., Lee, J., and Prett, D. (1993). *Model predictive control*. Prentice Hall Englewood Cliffs, NJ.

Rushby, J. (2002). Using model checking to detect automation surprises. *Reliability Engineering and System Safety*, 75(2), 167–177.

Sadigh, D., Dragan, A., Sastry, S.S., and Seshia, S.A. (2017). Active preference-based learning of reward functions. In *Proceedings of the Robotics: Science and Systems Conference (RSS)*.

Sadigh, D., Driggs-Campbell, K., Puggelli, A., Li, W., Shia, V., Bajcsy, R., Sangiovanni-Vincentelli, A.L., Sastry, S.S., and Seshia, S.A. (2014). Data-driven probabilistic modeling and verification of human driver behavior. *Formal Verification and Modeling in Human-Machine Systems*.

Sadigh, D., Sastry, S.S., Seshia, S.A., and Dragan, A. (2016a). Information gathering actions over human internal state. In *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, 66–73. IEEE.

Sadigh, D., Sastry, S.S., Seshia, S.A., and Dragan, A. (2016b). Planning for autonomous cars that leverages effects on human actions. In *Proceedings of the Robotics: Science and Systems Conference (RSS)*.

Shia, V.A., Gao, Y., Vasudevan, R., Campbell, K.D., Lin, T., Borrelli, F., and Bajcsy, R. (2014). Semiautonomous vehicular control using driver modeling. *IEEE Transactions on Intelligent Transportation Systems*, 15(6), 2696–2709.

Vasudevan, R., Shia, V., Gao, Y., Cervera-Navarro, R., Bajcsy, R., and Borrelli, F. (2012). Safe semi-autonomous control with enhanced driver modeling. In *American Control Conference (ACC), 2012*, 2896–2903. IEEE.

Wächter, A. and Biegler, L.T. (2006). On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical programming*, 106(1), 25–57.

Ziebart, B.D. (2010). *Modeling purposeful adaptive behavior with the principle of maximum causal entropy*. Ph.D. thesis, Carnegie Mellon University.

Ziebart, B.D., Maas, A.L., Bagnell, J.A., and Dey, A.K. (2008). Maximum entropy inverse reinforcement learning. In *AAAI*, 1433–1438.