

Active Constraint Learning in High Dimensions from Demonstrations

Zheng Qiu

School of Mechanical Engineering, Georgia Institute of Technology

ZQIU67@GATECH.EDU

Chih-Yuan Chiu

School of Electrical and Computer Engineering, Georgia Institute of Technology

CYC@GATECH.EDU

Glen Chou

Schools of Cybersecurity & Privacy and Aerospace Engineering, Georgia Institute of Technology

CHOU@GATECH.EDU

Editors: G. Sukhatme, L. Lindemann, S. Tu, A. Wierman, N. Atanasov

Abstract

We present an iterative active constraint learning (ACL) algorithm, within the learning from demonstrations (LfD) paradigm, which intelligently solicits informative demonstration trajectories for inferring an unknown constraint in the demonstrator’s environment. Our approach iteratively trains a Gaussian process (GP) on the available demonstration dataset to represent the unknown constraints, uses the resulting GP posterior to query start/goal states, and generates informative demonstrations which are added to the dataset. Across simulation and hardware experiments using high-dimensional nonlinear dynamics and unknown nonlinear constraints, our method outperforms a baseline, random-sampling based method at accurately performing constraint inference from an iteratively generated set of sparse but informative demonstrations.

Keywords: Learning from demonstration, Constraint inference, Gaussian Processes, Robot safety.

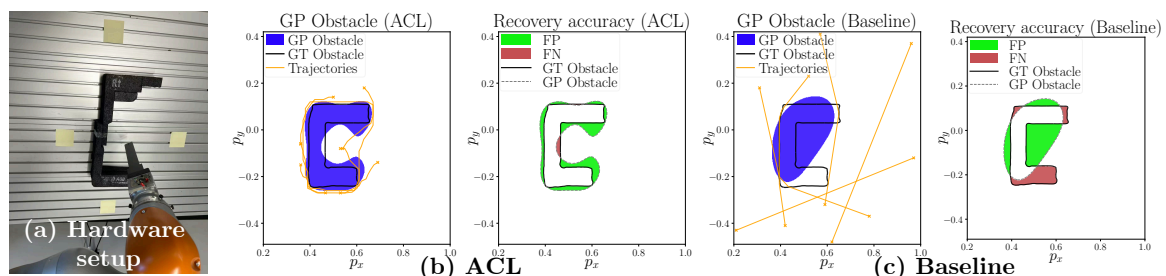


Figure 1: Given demonstrations (orange) generated via a 7-DOF robot arm, our GP-ACL algorithm recovers a nonlinear obstacle set with fewer instances of false positive (FP, in green) and false negative (FN, in red) errors, compared to a random-sampling baseline method.

1. Introduction

Recently, learning from demonstrations (LfD) via inverse optimal control (IOC) has emerged as a versatile framework for empowering robots to infer the constraints underlying an expert demonstrator’s actions (Chou et al., 2022; Menner et al., 2021; Chou et al., 2020b; Armesto et al., 2017; Papadimitriou and Li, 2023), with existing methods proving effective at inferring unknown constraints from a given set of approximately locally-optimal demonstrations. However, current approaches implicitly assume that the demonstrations supplied for training are generated in a manner agnostic to the downstream constraint learning task, and thus the supplied data may not always be informative for reducing constraint uncertainty. As such, existing constraint inference approaches may be data-inefficient, requiring a large number of demonstrations to accurately learn unknown constraints.

To overcome this limitation, we present an iterative *active* constraint learning algorithm built upon the Gaussian process (GP)-based constraint learning framework in Chou et al. (2022), which

uses the Karush-Kuhn-Tucker (KKT) optimality conditions of the provided demonstrations to train nonparametric GP-based constraint representations. At each iteration, our algorithm trains a Gaussian Process (GP) on a given dataset of locally-optimal demonstrations to represent the unknown constraint as a GP posterior. Although active learning methods have been widely explored in the context of *cost* inference, to our knowledge, our work provides the first framework for active *constraint* inference over continuous state spaces. Our contributions are:

1. Leveraging the GP-based constraint learning framework in Chou et al. (2022), we sample multiple GP posterior estimates of an unknown constraint given a demonstration dataset \mathcal{D} . The generation of multiple GP posterior samples provides a more complete description of the constraint information encoded in \mathcal{D} , as well as the remaining constraint uncertainty that cannot be resolved using \mathcal{D} .
2. We present a GP-based active constraint learning algorithm, GP-ACL (Alg. 1), which iteratively queries start/goal constraint states to induce the generation of informative demonstrations that reduce constraint uncertainty.
3. We evaluate our GP-ACL algorithm by inferring high-dimensional, nonlinear constraints from the generated demonstrations, both via 4D unicycle and 12D quadcopter dynamics in simulation, and via a 7-DOF robot arm hardware platform.

Related Works Existing LfD-based methods have enabled constraint learning via both IOC (Chou et al. (2020a, 2022); Armesto et al. (2017); Papadimitriou et al. (2022); Menner et al. (2021)) and inverse reinforcement learning frameworks (McPherson et al. (2021); Stocking et al. (2022); Papadimitriou and Brown (2024); Singh et al. (2018)). Specifically, Chou et al. (2020a, 2022); Papadimitriou and Li (2023) use KKT optimality conditions corresponding to the provided demonstrations to formulate inverse optimization problems, from which constraint information can then be extracted. In particular, our work builds upon and is most similar to Chou et al. (2022), which likewise embeds the KKT optimality conditions for the provided demonstrations into a Gaussian process (GP) framework to represent unknown constraints. However, whereas Chou et al. (2022) only computes the mean and covariance functions of a trained GP posterior for constraint inference, we additionally sample GP posterior samples as surrogate functions for the unknown constraint map, to facilitate the efficient generation of informative demonstrations. Moreover, while existing constraint learning methods can enable downstream planning that is robust to epistemic uncertainty (Chou et al., 2022, 2021), they do not actively seek to reduce uncertainty through soliciting informative demonstrations. In contrast, our GP-ACL algorithm explicitly actively generates demonstrations guided by the downstream constraint learning objective, a capability not considered in prior work.

Prior work has also considered the problem of active *intent* inference, with the aim of extracting the unknown reward or cost of an expert demonstrator from a provided set of trajectory demonstrations. In particular, methods have been developed to achieve active information gathering (Sadigh et al., 2016, 2018; Li et al., 2025b), intent demonstration (Li et al., 2024), and uncertainty reduction (Mesbah, 2018; Hu and Fisac, 2023) for human-robot interaction tasks. Meanwhile, Akrouf et al. (2012); Fang et al. (2017); Lee et al. (2021) devise active reward learning methods for reinforcement learning (RL). Our work likewise considers the purposeful generation of demonstrations that are maximally informative with respect to a downstream inference task. However, unlike the works listed above, our GP-ACL algorithm leverages demonstration data to recover unknown *constraints*, rather than unknown intent, rewards, or costs.

Finally, our methods are related to recently developed RL-based (Papadimitriou et al. (2022)) and GP-based (Li et al. (2025a)) active constraint learning methods. Unlike these works, however, our method is capable of inferring unknown high-dimensional constraints in continuous and infinite state and constraint spaces and does not require both constraint-satisfying and constraint-violating behavior to guide learning, which is important for safety-critical robotics applications.

2. Preliminaries and Problem Formulation

2.1. Demonstration Generation and KKT Optimality Conditions

By a *demonstration*, we refer to a state-control trajectory $\xi := (x_1(\xi), \dots, x_T(\xi), u_1(\xi), \dots, u_T(\xi)) \in \mathbb{R}^{(n+n_i)T}$ of length $T \in \mathbb{N}$, where $x_t(\xi) \in \mathbb{R}^n$ and $u_t(\xi) \in \mathbb{R}^{n_i}$ respectively denote the state system and control vector identified with the demonstration ξ at each time $t \in [T] := \{1, \dots, T\}$. As in Chou et al. (2022, Sec. III-A), we assume that each demonstration is a locally-optimal solution to the constrained optimization problem described below (Prob. 1). First, let $c : \mathbb{R}^{(n+n_i)T} \rightarrow \mathbb{R}$, $\mathbf{g}_k : \mathbb{R}^{(n+n_i)T} \rightarrow \mathbb{R}^{N_k^{\text{ineq}}}$, and $\mathbf{h}_k : \mathbb{R}^{(n+n_i)T} \rightarrow \mathbb{R}^{N_k^{\text{eq}}}$ respectively encode a (possibly non-convex) cost function, as well as a set of *known* inequality and equality constraints. In our work, we focus on smoothness-based costs of the form $c(\xi) := \sum_{t=1}^{T-1} \|x_{t+1} - x_t\|_2^2$; similar costs are often used in the constraint learning literature to encode trajectory length minimization (Chou et al. (2022); Papadimitriou and Li (2023)). Moreover, the known equality constraint $\mathbf{h}_k(\cdot) = 0$ encodes a set of deterministic, nonlinear dynamics $x_{t+1} = f_t(x_t, u_t), \forall t \in [T]$, as well as constraints on the initial and final system states, which we describe in more detail in Sec. 2.4.

Next, to formulate constraints unknown to the learner, let $\phi_{\text{sep}} : \mathbb{R}^n \rightarrow \mathbb{R}^{n_c}$ denote a map from each system state x to a *constraint state* $\phi_{\text{sep}}(x) \in \mathbb{R}^{n_c}$, at which the constraint satisfaction of the system state x is then evaluated¹, and let $\phi : \mathbb{R}^{nT} \rightarrow \mathbb{R}^{n_c T}$ be given by $\phi(\xi) := (\phi_{\text{sep}}(x_1(\xi)), \dots, \phi_{\text{sep}}(x_T(\xi)))$. For each $t \in [T]$, we formulate the unknown inequality constraints below, where $g_{1,\gamma_k}^*, \dots, g_{M,\gamma_k}^* : \mathbb{R}^{n_c} \rightarrow \mathbb{R}$ encode scalar-valued constraints $g_{m,\gamma_k}^*(\cdot) \leq 0$. For each $t \in [T]$, we can write “ $g_{m,\gamma_k}^*(\phi_{\text{sep}}(x_t(\xi))) \leq 0, \forall m \in [M]$ ” in the more compact form of “ $g_{\gamma_k}^*(x_t) \leq 0$ ”, by defining $g_{\gamma_k}^* : \mathbb{R}^{n_c} \rightarrow \mathbb{R}$ via $g_{\gamma_k}^*(\kappa) := \max\{g_{1,\gamma_k}^*(\kappa), \dots, g_{M,\gamma_k}^*(\kappa)\}$ for each *constraint state* $\kappa \in \mathbb{R}^{n_c}$. Moreover, we can stack the constraints $g_{\gamma_k}^*(\phi_{\text{sep}}(x_t(\xi))) \leq 0$ across times $t \in [T]$ as $\mathbf{g}_{\gamma_k}^*(\phi(\xi)) \leq 0$, by defining $\mathbf{g}_{\gamma_k}^* : \mathbb{R}^{n_c T} \rightarrow \mathbb{R}^T$ via $\mathbf{g}_{\gamma_k}^*(\phi(\xi)) := (g_{\gamma_k}^*(\phi_{\text{sep}}(x_1(\xi))), \dots, g_{\gamma_k}^*(\phi_{\text{sep}}(x_T(\xi))))$ for each demonstration $\xi \in \mathbb{R}^{(n+n_i)T}$.

Finally, we write the demonstrator’s trajectory generation problem as follows.

Problem 1 (Demonstrator’s forward problem)

$$\min_{\xi} \quad c(\xi) \tag{1a}$$

$$\text{s.t.} \quad \mathbf{h}_k(\xi) = \mathbf{0}, \quad \mathbf{g}_k(\xi) \leq \mathbf{0}, \quad \mathbf{g}_{\gamma_k}^*(\phi(\xi)) \leq \mathbf{0}, \tag{1b}$$

For each demonstration ξ , which forms a locally-optimal solution of Prob. 1, there must exist Lagrange multipliers $\lambda_k \in \mathbb{R}^{N_k^{\text{ineq}}}$, $\lambda_{\gamma_k} \in \mathbb{R}^T$, and $\nu \in \mathbb{R}^{N_k^{\text{eq}}}$ such that the following KKT optimality conditions, denoted below by $(\xi, \lambda_k, \lambda_{\gamma_k}, \nu) \in \text{KKT}$, hold:

$$\mathbf{g}_{\gamma_k}^*(\phi(\xi)) \leq \mathbf{0}, \tag{2a}$$

$$\lambda_k \geq \mathbf{0}, \quad \lambda_{\gamma_k} \geq \mathbf{0}, \tag{2b}$$

$$\lambda_k \odot \mathbf{g}_k(\xi) = \mathbf{0}, \quad \lambda_{\gamma_k} \odot \mathbf{g}_{\gamma_k}^*(\xi) = \mathbf{0}, \tag{2c}$$

$$\nabla_{\xi} c(\xi) + \lambda_k^{\top} \nabla_{\xi} \mathbf{g}_k^*(\xi) + \lambda_{\gamma_k}^{\top} \nabla_{\xi} \mathbf{g}_{\gamma_k}^*(\phi(\xi)) + \nu^{\top} \nabla_{\xi} \mathbf{h}_k(\xi) = \mathbf{0}, \tag{2d}$$

1. Our formulation readily generalizes to settings in which some unknown constraints depend on control inputs.

where \odot denotes element-wise multiplication, and for each differentiable map $\mathbf{f} : \mathbb{R}^n \rightarrow \mathbb{R}^m$, we define the gradient of f by $[\nabla_x f]_{ij} := \frac{\partial f_i}{\partial x_j}$ for each $i \in [m]$ and $j \in [n]$. Above, (2a), (2b), (2c), and (2d) encode primal and dual feasibility, complementary slackness, and stationarity conditions, respectively. We denote the stationarity residual, i.e., the left-hand-side of (2d), by $\mathbf{s}(\xi, \boldsymbol{\lambda}_k, \boldsymbol{\lambda}_{\neg k}, \boldsymbol{\nu}) \in \mathbb{R}^{(n+n_i)T}$, and the component of $\mathbf{s}(\xi, \boldsymbol{\lambda}_k, \boldsymbol{\lambda}_{\neg k}, \boldsymbol{\nu})$ corresponding to partial gradients with respect to system state x_t (resp., control u_t) by $\mathbf{s}_{x_t}(\xi, \boldsymbol{\lambda}_k, \boldsymbol{\lambda}_{\neg k}, \boldsymbol{\nu}) \in \mathbb{R}^n$ (resp., $\mathbf{s}_{u_t}(\xi, \boldsymbol{\lambda}_k, \boldsymbol{\lambda}_{\neg k}, \boldsymbol{\nu}) \in \mathbb{R}^{n_i}$).

2.2. Constraint Tightness and Constraint Gradient Extraction

Suppose we are given a collection of D demonstrations $S_D := \{\xi_d : d \in [D]\}$ generated by solving Prob. 1 to local optimality, with corresponding Lagrange multipliers $\boldsymbol{\lambda}_{d,k}$, $\boldsymbol{\lambda}_{d,\neg k}$, and $\boldsymbol{\nu}_d$ for each $d \in [D]$. Constraint information can be extracted from each ξ_d by examining its *tight* system states (if any), i.e., system states $x_t(\xi)$ at which $g_k^*(\phi_{\text{sep}}(x_t(\xi))) = 0$, as well as the constraint gradient values $\nabla_{x_t} g_k^*(\phi_{\text{sep}}(x_t(\xi)))$ at such tight system states. Methods for both extracting tight system states and computing the corresponding gradient values were first presented in Sec. IV-A in Chou et al. (2022); for completeness, we review details of these methods in App. A.1.

2.3. Training Gaussian Process (GP)-based Constraint Representations

To learn a Gaussian Process (GP)-based representation for the unknown constraint function $g_k^*(\cdot)$, we collect the constraint states and estimated constraint gradients, across the tight timesteps $t \in \mathbf{t}_{\text{tight}}(\xi_d)$ of each demonstration $d \in [D]$, into the data sets \mathcal{D}_κ and \mathcal{D}_∇ , as defined below. Here, for each $d \in [D]$ and $t \in [T]$, $w_{d,t} \in \mathbb{R}^{1 \times n}$ denotes an estimate for the unknown constraint gradient $\nabla_{x_t} g_k^*(\phi_{\text{sep}}(x_t(\xi_d)))$, taken with respect to the system state x_t ; we defer a thorough discussion of the computation of $w_{d,t}$ to Prob. 8 in App. A.1:

$$\mathcal{D}_\kappa := \{\phi_{\text{sep}}(x_t(\xi_d)) : d \in [D], t \in \mathbf{t}_{\text{tight}}(\xi_d)\}, \quad (3)$$

$$\mathcal{D}_\nabla := \{w_{d,t} : d \in [D], t \in \mathbf{t}_{\text{tight}}(\xi_d)\} \quad (4)$$

Since $g_k^*(\phi_{\text{sep}}(x_t(\xi_d))) = 0 \forall t \in \mathbf{t}_{\text{tight}}(\xi_d)$, we append $\mathcal{D}_g := \{\mathbf{0} \in \mathbb{R}^{N_{\text{tight}}}\}$, where $N_{\text{tight}} := \sum_{d=1}^D |\mathbf{t}_{\text{tight}}(\xi_d)|$, into our training data ($|\cdot|$ denotes set cardinality). Let $\mathbf{X} := \mathcal{D}_\kappa$ and $\mathbf{Y} := (\mathcal{D}_g, \mathcal{D}_\nabla)$ respectively be input and output training datasets for our GP model, set $\mathcal{D} := (\mathbf{X}, \mathbf{Y})$, and optimize GP hyperparameters via the marginal log likelihood (Rasmussen and Williams (2006)). Our GP training process emulates (Chou et al., 2022, Sec. IV-B). The resulting posterior $(\tilde{g}|\mathcal{D})(\cdot)$ of g_k^* has posterior mean $\mathbb{E}[\tilde{g}(\cdot)|\mathcal{D}]$ and covariance $\text{cov}[\tilde{g}(\cdot)|\mathcal{D}]$ (see (15)). A plausible constraint $\hat{g}(\cdot)$ can be sampled from this posterior via random Fourier features. For details, see App. A.2.

2.4. Problem Statement and Formulation

Suppose we are given a demonstration dataset S_D and a corresponding GP posterior $(\tilde{g}|\mathcal{D})(\cdot)$ describing our belief of the unknown, ground truth constraint $g_k^*(\cdot)$. We seek start (κ_s) and goal (κ_g) constraint states such that locally-optimal demonstrations ξ generated while adhering to the start and goal constraints $\phi_{\text{sep}}(x_1(\xi)) = \kappa_s$ and $\phi_{\text{sep}}(x_T(\xi)) = \kappa_g$ maximally reduce the remaining uncertainty over the GP posterior $(\tilde{g}|\mathcal{D})(\cdot)$. Concretely, we formulate our problem as follows:

Problem 2 (Active Constraint Learning (Idealized))

$$\max_{\kappa_s, \kappa_g, \xi, \boldsymbol{\lambda}_k, \boldsymbol{\lambda}_{\neg k}, \boldsymbol{\nu}} \cdot \sum_{t \in \mathbf{t}_{\text{tight}}(\xi)} \text{cov}[\tilde{g}(\phi_{\text{sep}}(x_t(\xi)))|\mathcal{D}] \quad (5a)$$

$$\text{s.t.} \quad \phi_{\text{sep}}(x_1(\xi)) = \kappa_s, \quad \phi_{\text{sep}}(x_T(\xi)) = \kappa_g, \quad (\xi, \boldsymbol{\lambda}_k, \boldsymbol{\lambda}_{\neg k}, \boldsymbol{\nu}) \in \text{KKT}(S_D). \quad (5b)$$

Above, given a candidate demonstration ξ , the objective (5a) evaluates the covariance of the GP posterior $(\tilde{g}|\mathcal{D})(\cdot)$ at the robustly identified tight states $\{x_t(\xi) : t \in \mathbf{t}_{\text{tight}}(\xi)\}$ of ξ . Since the covariance functions of GPs serve as uncertainty measures, (5a) captures the degree to which a candidate ξ traverses regions of high constraint uncertainty with respect to the GP posterior $(\tilde{g}|\mathcal{D})(\cdot)$. Meanwhile, (5b) enforce the start/goal constraints and the local optimality of ξ (see Prob. 1).

A constraint learner who attempts to directly solve Prob. 2 faces several challenges. First, the KKT conditions $\text{KKT}(S_D)$ appearing in (5b) depend on knowledge of the unknown constraint $g_k^*(\cdot)$ (see (2)), which the constraint learner *a priori* lacks. Moreover, for each candidate demonstration ξ , the tight timesteps $\mathbf{t}_{\text{tight}}(\xi)$ appearing in (5a) can only be computed by solving Prob. 7 as an inner optimization loop, which renders Prob. 2 computationally intractable. To overcome these challenges, in Sec. 3, we decompose Prob. 2 into sub-problems that allow the recovery of start/goal constraint states which are approximately maximally-informative for the constraint learning task.

3. Methods

Below, Sec. 3.1 describes optimization routines for obtaining start/goal constraint states which induce approximately maximally informative demonstrations, to bypass the challenges of directly tackling Prob. 2, as identified in Sec. 2.4. The prescribed optimization steps are then synthesized in Sec. 3.2 into our iterative active constraint learning algorithm (Alg. 1), which repeatedly queries start and goal states from which the demonstrator is likely to generate state-control trajectories which maximally reduce the remaining uncertainty over the unknown constraints.

3.1. Optimization Routines for Approximately Solving Prob. 2

Suppose, as in Prob. 2, that we are given a demonstration set S_D and corresponding GP posterior $(\tilde{g}|\mathcal{D})(\cdot)$ and P GP posterior samples $\{\hat{g}_p(\cdot)\}_{p \in [P]}$. We aim to select suitable start/goal constraint states for constraint learning from each GP posterior sample $\hat{g}_p(\cdot)$. First, we compute the *maximally informative constraint state* $\kappa_{\text{MI},p}$, defined as the constraint state which maximizes the covariance of the GP posterior $(\tilde{g}|\mathcal{D})(\cdot)$ while remaining safe with respect to $\hat{g}_p(\cdot)$ (Prob. 3). We then approximate Prob. 2 as the problem of searching for start/goal constraint states, using each $\kappa_{\text{MI},p}$ and corresponding gradient $\nabla \hat{g}_p(\kappa_{\text{MI},p})$, which induce tight demonstrations against the unknown constraint at $\kappa_{\text{MI},p}$, and thus provide information about the constraint shape near $\kappa_{\text{MI},p}$. Concretely, we present two methods for computing start/goal constraint states, as codified in Probs. 4-5 and Prob. 6 below, which are tailored respectively to the settings in which the avoid set $\mathcal{A} := \{\kappa \in \mathbb{R}^{n_c} : g_k^*(\kappa) > 0\}$ (i) is locally-convex near $\kappa_{\text{MI},p}$, or (ii) is not. Since the constraint learner lacks *a priori* knowledge of the geometry of \mathcal{A} , our GP-ACL approach prescribes the application of *both* methods to generate two start/goal constraint state pairs.

Finding Maximally-Informative Constraint States $\{\kappa_{\text{MI},p}\}_{p \in [P]}$ We begin by computing, for each GP posterior sample $\hat{g}_p(\cdot)$, a maximally-informative constraint state $\kappa_{\text{MI},p}$ which maximizes the covariance of the GP posterior $(\tilde{g}|\mathcal{D})(\cdot)$ while remaining tight with respect to $\hat{g}_p(\cdot)$.

Problem 3 (Computing Maximally-Informative Constraint States $\{\kappa_{\text{MI},p}\}_{p \in [P]}$)

$$\kappa_{\text{MI},p} \in \arg \max_{\kappa \in \mathbb{R}^{n_c}} \cdot \quad \text{cov}[\tilde{g}(\kappa)|\mathcal{D}] \quad (6a)$$

$$\text{s.t.} \quad \hat{g}_p(\kappa) = 0. \quad (6b)$$

Below, we present separate schemes for generating start/goal constraint states under the distinct scenarios in which the avoid set \mathcal{A} either is locally convex near $\kappa_{\text{MI},p}$ (Fig. 2a) or is not (Fig. 2b).

Optimizing Start/Goal Constraint States in a Hyperplane $\mathcal{H}(\eta)$ Orthogonal to $\nabla \hat{g}_p(\kappa_{\text{MI},p})$

We begin by formulating a search method for start/goal constraints that is adapted for the setting in which the avoid set \mathcal{A} is locally-convex near $\kappa_{\text{MI},p}$, and thus the boundary of \mathcal{A} curves away from $\kappa_{\text{MI},p}$ in the direction of $\nabla \hat{g}_p(\kappa_{\text{MI},p})$ (see Fig. 2a). We aim to induce demonstrations that are taut against \mathcal{A} (and thus reveal information about \mathcal{A}) near $\kappa_{\text{MI},p}$. To this end, we first fix a constraint state $\kappa_p(\eta)$ that is slightly offset from $\kappa_{\text{MI},p}$ in the direction of $\nabla \hat{g}_p(\kappa_{\text{MI},p})$, with the degree of offset measured by the step size η . The purpose of defining a constraint state $\kappa_p(\eta)$ offset from $\kappa_{\text{MI},p}$ is to increase the likelihood that the resulting demonstration is tight against the avoid set \mathcal{A} at $\kappa_{\text{MI},p}$, i.e., the true constraint g_k^* is active at $\kappa_{\text{MI},p}$, for a constraint that is locally-convex around $\kappa_{\text{MI},p}$ (see Fig. 2a). We then search for start/goal constraint states $(\kappa_{s,p,\perp}, \kappa_{g,p,\perp})$ outside \mathcal{A} but within a hyperplane $\mathcal{H}_p(\eta)$ which is orthogonal to $\nabla \hat{g}_p(\kappa_{\text{MI},p})$ and passes through $\kappa_p(\eta)$ (see Fig. 2a). As illustrated in Fig. 2a, the demonstration trajectory generated from $(\kappa_{s,p,\perp}, \kappa_{g,p,\perp})$ is likely to curve away from $\kappa_{\text{MI},p}$ in the direction of $\nabla \hat{g}_p(\kappa_{\text{MI},p})$, similar to the boundary of \mathcal{A} near $\nabla \hat{g}_p(\kappa_{\text{MI},p})$, and yield a tight point at $\kappa_{\text{MI},p}$ at which the true constraint g_k^* is active.

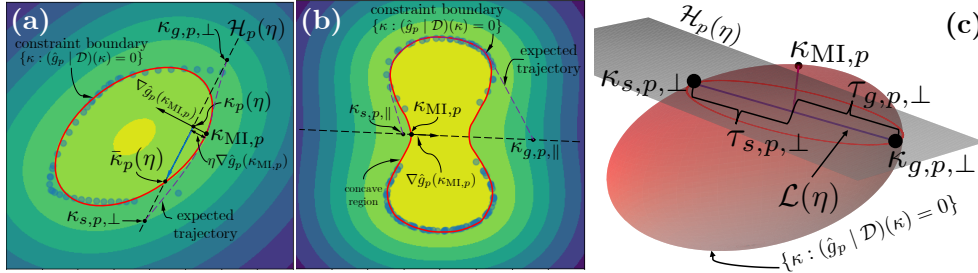


Figure 2: (a) Our method for finding start/goal constraint states that induce tight demonstrations. Blue dots represent tight points (from demonstrations) used to train the GP constraint representation (see Sec. 2.2, 2.3, and A.1). To handle locally-convex constraint boundaries, we search along a hyperplane $\mathcal{H}(\eta)$ orthogonal to the gradient $\nabla \hat{g}_p(\kappa_{\text{MI},p})$. (b) To handle locally-concave constraint boundaries, we search along $\nabla \hat{g}_p(\kappa_{\text{MI},p})$. (c) Visualization of the locally-convex case in 3D.

Concretely, we first fix a suitably small step size η , and define:

$$\kappa_p(\eta) := \kappa_{\text{MI},p} + \eta \nabla \hat{g}_p(\kappa_{\text{MI},p}), \quad \forall p \in [P], \quad (7)$$

$$\mathcal{H}_p(\eta) := \{ \kappa \in \mathbb{R}^{n_c} : \nabla \hat{g}_p(\kappa_{\text{MI},p})(\kappa - \kappa_p(\eta)) = 0 \}, \quad \forall p \in [P]. \quad (8)$$

To fix a search direction within $\mathcal{H}(\eta)$ starting from $\kappa_p(\eta)$, we aim to find a constraint state $\bar{\kappa}_p(\eta)$ in $\mathcal{H}(\eta)$ that is of maximum possible distance from $\kappa_p(\eta)$ while remaining *unsafe* with respect to the GP posterior sample $\hat{g}_p(\cdot)$ (Prob. 4). Intuitively, $\bar{\kappa}_p(\eta)$ is a constraint state on the boundary of \mathcal{A} that lies on $\mathcal{H}_p(\eta)$ which we aim to locate, to obtain a search direction (in the form of $\kappa_p(\eta) - \bar{\kappa}_p(\eta)$) along which safe start/goal constraint states can be located.

Problem 4 (Identifying a Search Direction $\bar{\kappa}_p(\eta) - \kappa_p(\eta)$ in $\mathcal{H}(\eta)$)

$$\bar{\kappa}_p(\eta) \in \arg \max_{\kappa \in \mathbb{R}^{n_c}} \cdot \|\kappa - \kappa_p(\eta)\|_2 \quad (9a)$$

$$\text{s.t.} \quad \kappa \in \mathcal{H}(\eta), \quad \hat{g}_p(\kappa) > 0. \quad (9b)$$

We then search for start/goal constraint states along the line $\mathcal{L}(\eta) := \{ \kappa_p(\eta) + t(\bar{\kappa}_p(\eta) - \kappa_p(\eta)) : t \in \mathbb{R} \}$ on $\mathcal{H}(\eta)$ (Prob. 5), which identifies, among all vectorial directions in $\mathcal{H}(\eta)$ starting from $\kappa_p(\eta)$, the direction along which constraint states remain unsafe over the greatest distance (Fig. 2c). Since demonstrators aim to minimize path length while ensuring feasibility, start/goal constraint states on $\mathcal{L}(\eta)$ outside the avoid set \mathcal{A} are likely to induce demonstrations that are taut against \mathcal{A} .

Concretely, for each $p \in [P]$, we search for start/goal constraint states on $\mathcal{L}(\eta)$ that are as close to $\kappa_p(\eta)$ as possible while remaining safe. To encode safety, we consider constraint states κ that are (i) strictly safe with respect to $\hat{g}_p(\cdot)$ by a margin $\delta > 0$, and (ii) safe with respect to the GP posterior $(\tilde{g}|\mathcal{D})(\cdot)$ with probability at least $\beta > 0$, where both δ and β are algorithm design parameters. Mathematically, the two safety conditions described above are given by:

$$\hat{g}_p(\kappa) \leq -\delta, \quad \Phi\left(\frac{\mathbb{E}[\tilde{g}(\kappa)|\mathcal{D}]}{\sqrt{\text{cov}[\tilde{g}(\kappa)|\mathcal{D}]}}\right) \geq \beta, \quad (10)$$

where $\Phi(\cdot)$ is the cumulative distribution function (CDF) of the unit Gaussian distribution. We formulate Prob. 5 below to compute optimal start and goal constraint states, denoted respectively by $\kappa_{s,p,\perp}$ and $\kappa_{g,p,\perp}$, along the two halves of $\mathcal{L}(\eta)$ corresponding to the selection of negative or positive values of the scale parameter τ , which measures distance from the unsafe constraint state $\kappa_p(\eta)$. We choose τ^2 as our objective in (11a) to compel the start/goal constraint states to be close to $\kappa_p(\eta)$ while remaining probabilistically safe in the sense of (10), to increase the likelihood of generating demonstrations that are close to, and tight against, the avoid set \mathcal{A} (see Fig. 2a).

Problem 5 (Optimizing Start/Goal Constraint States in $\mathcal{H}(\eta)$ Along $\bar{\kappa}_p(\eta) - \kappa_p(\eta)$)

$$\begin{aligned} & (\tau_{s,p,\perp}, \kappa_{s,p,\perp}) \text{ OR } (\tau_{g,p,\perp}, \kappa_{g,p,\perp}) \\ \in \arg \min_{\substack{\tau \in \mathbb{R} \\ \kappa \in \mathbb{R}^{n_c}}} \tau^2 \end{aligned} \quad (11a)$$

$$\text{s.t. } \kappa = \kappa_p(\eta) + \tau(\bar{\kappa}_p(\eta) - \kappa_p(\eta)), \quad \kappa \text{ satisfies (10),} \quad \tau < 0 \text{ OR } \tau > 0. \quad (11b)$$

Remark 1 While numerically solving Prob. 5, we often encode the constraints (10) as penalties in the cost (11a) with large weights, to bypass the issue that initializations of κ may fail to satisfy (10).

Optimizing Start/Goal Constraint States Along $\nabla \hat{g}_p(\kappa_{\text{MI},p})$ If the true constraint set were in fact *not* locally convex near $\kappa_{\text{MI},p}$, the boundary of \mathcal{A} may curve away from $\kappa_{\text{MI},p}$ in the direction of $-\nabla \hat{g}_p(\kappa_{\text{MI},p})$ (see Fig. 2b). In this case, start/goal constraints located by searching along $\pm \nabla \hat{g}_p(\kappa_{\text{MI},p})$ from $\kappa_{\text{MI},p}$, which we denote by $(\kappa_{s,p,\parallel}, \kappa_{g,p,\parallel})$, are likely to generate demonstrations that are tight against \mathcal{A} , since such demonstrations may also exhibit curvature away from $\kappa_{\text{MI},p}$ in the direction of $-\nabla \hat{g}_p(\kappa_{\text{MI},p})$, similar to the boundary of \mathcal{A} . An illustration is provided in Fig. 2b, and the computation of $(\kappa_{s,p,\parallel}, \kappa_{g,p,\parallel})$ is described in detail in Prob. 6. We note that Probs. 3-6 are efficiently solvable via the IPOPT solver (Wächter and Biegler, 2006) in Casadi (Andersson et al., 2019). Given that \mathcal{A} may or may not be locally-convex near $\kappa_{\text{MI},p}$, searching for start/goal constraint states in directions both orthogonal and parallel to the gradient $\nabla \hat{g}_p(\kappa_{\text{MI},p})$ (Probs. 4-6) increases the likelihood of generating demonstrations that are tight against the unknown constraint.

Concretely, Prob. 6 considers a variant of Prob. 5 which searches for start/goal states in the direction of $\nabla \hat{g}_p(\kappa_{\text{MI},p})$ from $\kappa_p(\eta)$, rather than along the direction $\bar{\kappa}_p(\eta) - \kappa_p(\eta)$, with the same aim of inducing tight demonstrations that are taut against the constraint boundary (see Fig. 2b).

Problem 6 (Optimizing Start/Goal Constraints Along $\nabla \hat{g}_p(\kappa_{\text{MI},p})$)

$$\begin{aligned} & (\tau_{s,p,\parallel}, \kappa_{s,p,\parallel}) \text{ OR } (\tau_{g,p,\parallel}, \kappa_{g,p,\parallel}) \\ \in \arg \min_{\tau \in \mathbb{R}, \kappa \in \mathbb{R}^{n_c}} \tau^2 \end{aligned} \quad (12a)$$

$$\text{s.t. } \kappa = \kappa_p(\eta) + \tau \nabla \hat{g}_p(\kappa_{\text{MI},p})^\top, \quad \kappa \text{ satisfies (10),} \quad \tau < 0 \text{ OR } \tau > 0. \quad (12b)$$

Algorithm 1: Gaussian Process-based Active Constraint Learning (GP-ACL) Algorithm.

Data: $N_{\text{iters}}, \alpha, n_\ell, \eta$, initial demonstration set $S_D^{(1)}, \delta, \beta$
Result: $S_D^{(N_{\text{iters}})}$

- 1: **for** $i = 1, \dots, N_{\text{iters}} - 1$ **do**
- 2: $(\tilde{g}^{(i)}|\mathcal{D}^{(i)})(\cdot), \{\hat{g}_p^{(i)}(\cdot) : p \in [P]\} \leftarrow$ Learn GP posterior and draw GP posterior samples
- 3: via Probs. 7-8, using the hyperparameters α and n_ℓ
- 4: $\{\kappa_{\text{MI},p}^{(i)}\}_{p \in [P]} \leftarrow$ Compute maximally-informative constraint states via Prob. 3
- 5: $\kappa_p(\eta), \mathcal{H}(\eta) \leftarrow$ Compute offset constraint state and hyperplane via (7) and (8)
- 6: $S_{\kappa,1}^{(i)} := \{(\kappa_{s,p,\perp}^{(i)}, \kappa_{g,p,\perp}^{(i)})\}_{p \in [P]} \leftarrow$ Compute start/goal constraint states via Probs. 4-5
(with δ, β)
- 7: $S_{\kappa,2}^{(i)} := \{(\kappa_{s,p,\parallel}^{(i)}, \kappa_{g,p,\parallel}^{(i)})\}_{p \in [P]} \leftarrow$ Compute start/goal constraint states via Prob. 6
- 8: $\tilde{S}_D^{(i)} \leftarrow \forall (\kappa_{s,j}^{(i)}, \kappa_{g,j}^{(i)}) \in S_{\kappa,1}^{(i)} \cup S_{\kappa,2}^{(i)}$, solve Prob. 1 over $\xi \in \mathbb{R}^{(n+n_i)T}$ subject to:
- 9: $\phi_{\text{sep}}(x_o(\xi)) = \kappa_{s,j}^{(i)}$ and $\phi_{\text{sep}}(x_T(\xi)) = \kappa_{g,j}^{(i)}$.
- 10: $S_D^{(i+1)} \leftarrow S_D^{(i)} \cup \tilde{S}_D^{(i)}$
- 11: **end for**

3.2. GP-ACL Algorithm

We present our algorithm for active constraint learning (GP-ACL) in Alg. 1. Concretely, given an available demonstration dataset $S_D^{(i)}$ at each iteration $i \in [N_{\text{iters}}]$, we solve Prob. 7 to extract tight points and perform GP learning via Prob. 8 to construct the stochastic GP posterior $(\tilde{g}^{(i)}|\mathcal{D})(\cdot)$ as an estimate of the unknown constraint function $g_k^*(\cdot)$ (Lines 2-3). From $(\tilde{g}^{(i)}|\mathcal{D})(\cdot)$, we compute the posterior mean $\mathbb{E}[\tilde{g}^{(i)}(\cdot)|\mathcal{D}]$ and randomly draw posterior samples $\{\hat{g}_p^{(i)}(\cdot)\}_{p \in [P]}$. Next, by solving Prob. 4-6, we query start and goal constraint states that are likely to compel the demonstrator to produce constraint-revealing trajectories (Lines 4-7). Finally, after new demonstrations are generated (see Prob. 1), we insert the newly generated demonstrations and their corresponding robustly identified constraint state and gradient values (see Sec. 2.3) into $S_D^{(i)}$ to form an updated dataset $S_D^{(i+1)}$, and proceed to the next iteration of our algorithm (Lines 8-10).

4. Experiments

To evaluate our GP-ACL algorithm, we perform constraint learning tasks on simulations using double integrator, 4D unicycle, 12D quadcopter, and 7-DOF robot arm dynamics (see (Murray et al., 1994, Ch. 2-4)), and on hardware platforms using a 7-DOF robot arm. Below, Sec. 4.1 introduces parameter settings shared across our experiments, while Sec. 4.2 presents a select subset of our experiment results. For additional experiments and experiment details, see App. B.

4.1. Experiment Setup

Our experiments involve the dynamics, constraints, and costs listed below. Given a state vector x_t , we use $p_t \in \mathbb{R}^3$ and $p_{x,t}, p_{y,t}$, and $p_{z,t} \in \mathbb{R}$ to respectively denote the overall 3D position vector and the x -, y -, and z - position coordinates encoded by x .

Dynamics models We infer constraints from demonstrations generated using 2D and 3D double-integrator, 4D unicycle, 12D quadcopter (Sabatino (2015)), and 7-DOF robot arm (Murray et al.

(1994)) dynamics. In our experiments, we discretize the above continuous-time dynamics at intervals of $\Delta t = 1$ and set a time horizon of $T = 30$, unless stated otherwise.

Constraints We consider eight unknown nonlinear constraints $\{g_{k,i}^* : i \in [8]\}$, each corresponding to an obstacle set \mathcal{A}_i that demonstrations must avoid. We define the constraint space for each \mathcal{A}_i to be either the configuration space of the robot arm, or in 2D/3D Cartesian coordinates for all other dynamics. For the definition of each constraint type, see Appendix B.1. In addition to the nonlinear constraints mentioned above, each demonstration must satisfy prescribed start/goal constraints.

Costs All demonstrations in the following experiments are generated via the smoothness cost $c(x) := \sum_{t=1}^{T-1} \|p_{t+1} - p_t\|_2^2$, which compels each demonstration to minimize the total distance traversed between the prescribed start/goal constraints while avoiding the prescribed obstacle sets.

Algorithm Implementation When running GP-ACL (Alg. 1), unless otherwise specified, we set $\alpha = 0.3$ and use $n_\ell = 1000$ random Fourier basis functions to train the GP posterior for constraint representation ((16) and Alg. 1, Lines 2-3), with $N_{\text{iters}} = 3$ and $\delta = 0.001$. We select $\beta = 0.55$ to compute start/goal constraint states $(\kappa_{s,p,\perp}, \kappa_{g,p,\perp})$ in Alg. 1, Line 6, and $\beta = 0.3$ and $\beta = 0.55$ to compute $\kappa_{s,p,\parallel}$ and $\kappa_{g,p,\parallel}$, respectively, on Line 7.

Metrics for Assessing Constraint Recovery Accuracy To evaluate the constraint recovery accuracy of our GP-ACL (resp., a random-sampling baseline) method, we randomly generate n_s samples in the constraint space and report the fraction γ_{ours} (resp., γ_{BL}) of sampled constraint states at which constraint satisfaction or violation was accurately predicted. For both methods, we also visualize constraint states corresponding to false positive (FP) and false negative (FN) errors, defined respectively as incidents in which safe constraint states are mistakenly marked as unsafe, and vice versa.

4.2. Experiment Results

Unicycle Simulations We evaluate our GP-ACL method by recovering three nonlinear constraints $g_{k,1}^*$, $g_{k,2}^*$, and $g_{k,3}^*$ (see App. B.1, Fig. 3). Here, we set $\alpha = 0.3$ when sampling GP posteriors. Across $n_s = 2500$ sampled constraint states, our GP-ACL algorithm accurately predicts the safeness or unsafeness of each sample with accuracy $\gamma_{\text{ours}} = 0.9996, 1.0, \text{ and } 0.9956$ for the constraints $g_{k,1}^*$, $g_{k,2}^*$ and $g_{k,3}^*$, respectively, while the random-sampling baseline yielded accuracies of only $\gamma_{\text{BL}} = 0.9788, 0.9912, \text{ and } 0.9492$, respectively. Although both the baseline and GP-ACL accurately classified most of the space in each obstacle set, our method incurred markedly lower misclassification rates near obstacle boundaries, resulting in higher constraint representation quality (Fig. 3).

Quadcopter Simulations Our GP-ACL algorithm also outperforms the baseline method in accurately recovering constraints from demonstrations of length $T = 20$ generated from high-dimensional quadcopter dynamics and the constraint $g_{k,5}^*$ (Fig. 10). Here, we set $\alpha = 0.3$ when sampling GP posteriors. Over $n_s = 27,000$ sampled constraint states, GP-ACL achieved higher accuracy ($\gamma_{\text{ours}} = 0.9919$) compared to the random sampling baseline ($\gamma_{\text{BL}} = 0.9423$). Our experiment results verify that, compared to random sampling, GP-ACL achieves superior constraint recovery accuracy when learning from demonstrations generated via high-dimensional nonlinear dynamics.

7-DOF Arm Simulations and Hardware Experiments Across constraint recovery tasks involving demonstrations of length $T = 20$ generated in simulation (resp., on hardware) using 7-DOF arm dynamics and the ellipse-shaped constraint $g_{k,7}^*$ (resp., using the physical obstacle visualized in Fig. 1), our GP-AL algorithm likewise achieves higher constraint inference accuracy compared to the baseline method. (We do not report γ_{ours} and γ_{BL} here, due to challenges inherent in sampling from a 7D constraint space.) For 7-DOF robot arm simulations and hardware experiments, we use

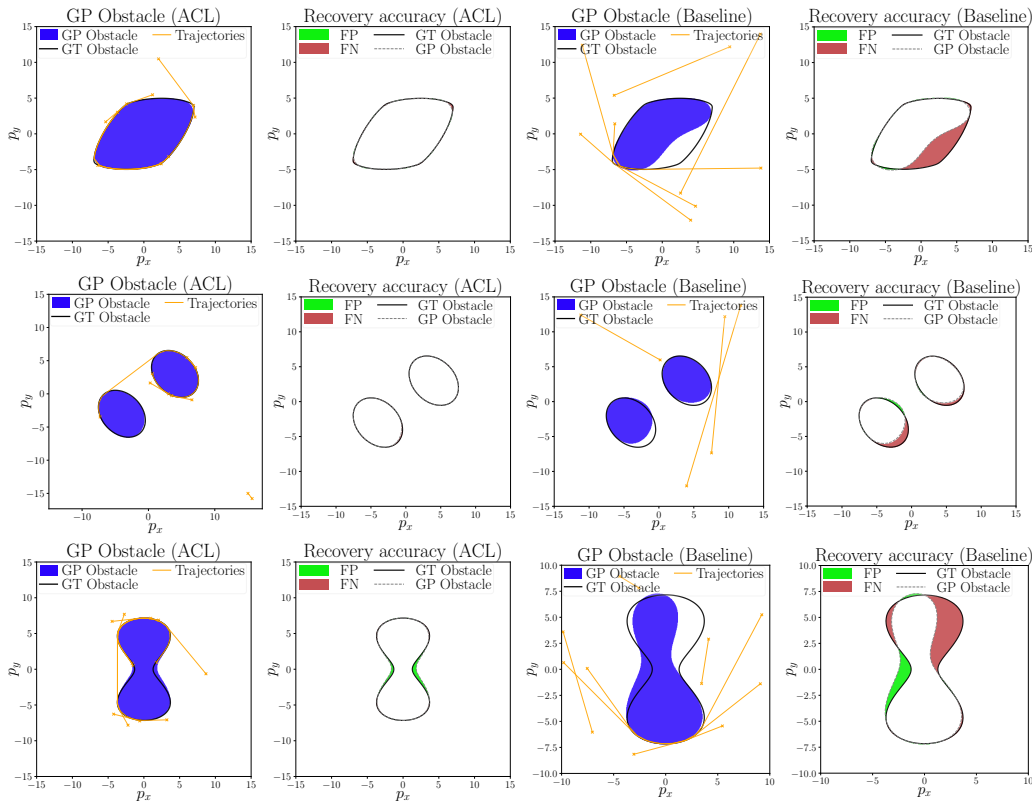


Figure 3: Our GP-ACL algorithm (left half) outperforms the random sampling baseline (right half) in accurately recovering constraints $g_{k,1}^*$ (top), $g_{k,2}^*$ (middle), and $g_{k,3}^*$ (bottom) from unicycle dynamics demonstrations, with fewer false positive (green) and false negative (red) errors.

$\delta = 0.1$ in our GP-ACL algorithm. Moreover, for hardware experiments, we set $\alpha = 0.1$ when sampling GP posteriors. Our experiment results illustrate that, when learning from either simulated or hardware demonstrations generated using high-dimensional robot arm dynamics, our GP-ACL algorithm achieves superior constraint recovery accuracy compared to the baseline approach.

Constraint Accuracy of GP-ACL vs. Baseline Fig. 11 plots constraint learning accuracy against iteration count for our GP-ACL method and the random-sampling baseline, when learning from demonstrations generated with unicycle, quadcopter, and 3D double integrator dynamics. GP-ACL consistently outperforms random sampling in per-iterate constraint learning accuracy.

5. Conclusion

This paper presents our Active Constraint Learning (GP-ACL) method for inferring unknown constraints through iterative, uncertainty-guided demonstration generation. In simulation and hardware experiments involving nonlinear, high-dimensional robot dynamics and non-convex constraints, GP-ACL successfully queries a small number of informative demonstrations to efficiently and accurately recover unknown constraints. In contrast to existing constraint inference techniques which learn from demonstrations generated without considerations of constraint uncertainty, our ACL method achieves higher constraint inference accuracy while learning from a substantially smaller, but more informative, demonstration dataset. Future work will extend GP-ACL to handle demonstrations generated using unknown or noise-corrupted dynamics [Chiu et al. \(2026\)](#), and to incorporate maximally informative waypoint states as side information.

References

- Riad Akrou, Marc Schoenauer, and Michèle Sebag. April: Active preference learning-based reinforcement learning. In Peter A. Flach, Tijn De Bie, and Nello Cristianini, editors, *Machine Learning and Knowledge Discovery in Databases*, pages 116–131, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.
- Joel A E Andersson, Joris Gillis, Greg Horn, James B Rawlings, and Moritz Diehl. CasADi: A Software Framework for Nonlinear Optimization and Optimal Control. *Mathematical Programming Computation*, 2019.
- Leopoldo Armesto, Jorren Bosga, Vladimir Ivan, and Sethu Vijayakumar. Efficient Learning of Constraints and Generic Null Space Policies. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1520–1526, 2017.
- Chih-Yuan Chiu, Zhouyu Zhang, and Glen Chou. Learning Constraints From Stochastic Partially-Observed Closed-Loop Demonstrations. *IEEE Control Systems Letters*, 10:43–48, 2026.
- Glen Chou, Necmiye Ozay, and Dmitry Berenson. Learning Constraints from Locally-optimal Demonstrations under Cost Function Uncertainty. *IEEE Robotics and Automation Letters*, 5(2): 3682–3690, 2020a.
- Glen Chou, Necmiye Ozay, and Dmitry Berenson. Learning Parametric Constraints in High Dimensions from Demonstrations. In *Conference on robot learning*, pages 1211–1230. PMLR, 2020b.
- Glen Chou, Dmitry Berenson, and Necmiye Ozay. Uncertainty-aware Constraint Learning for Adaptive Safe Motion Planning from Demonstrations. In *Conference on robot learning*, pages 1612–1639. PMLR, 2021.
- Glen Chou, Hao Wang, and Dmitry Berenson. Gaussian Process Constraint Learning for Scalable Chance-constrained Motion Planning from Demonstrations. *IEEE Robotics and Automation Letters*, 7(2):3827–3834, 2022.
- Meng Fang, Yuan Li, and Trevor Cohn. Learning how to Active Learn: A Deep Reinforcement Learning Approach. In Martha Palmer, Rebecca Hwa, and Sebastian Riedel, editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2017.
- Haimin Hu and Jaime F. Fisac. Active Uncertainty Reduction for Human-Robot Interaction: An Implicit Dual Control Approach. In *Springer, Springer Proceedings in Advanced Robotics*, pages 385–401. Springer Nature, 2023.
- Kimin Lee, Laura M. Smith, and P. Abbeel. PEBBLE: Feedback-Efficient Interactive Reinforcement Learning via Relabeling Experience and Unsupervised Pre-training. In *International Conference on Machine Learning*, 2021.
- Guoyan Li, Yujia Wang, Swastik Kar, and Xiaoning Jin. Bayesian Optimization with Active Constraint Learning for Advanced Manufacturing Process Design. *IISE Transactions*, 0(0):1–15, 2025a.

- Jingqi Li, Anand Siththaranjan, Somayeh Sojoudi, Claire Tomlin, and Andrea Bajcsy. Intent Demonstration in General-Sum Dynamic Games via Iterative Linear-Quadratic Approximations. *ArXiv*, abs/2402.10182, 2024.
- Zhongguo Li, Wen-Hua Chen, Jun Yang, and Cunjia Liu. Cooperative Active Learning-Based Dual Control for Exploration and Exploitation in Autonomous Search. *IEEE Transactions on Neural Networks and Learning Systems*, 36(2):2221–2233, 2025b.
- David McPherson, Kaylene Stocking, and Shankar Sastry. Maximum Likelihood Constraint Inference from Stochastic Demonstrations. In *CCTA*, 2021.
- Marcel Menner, Peter Worsnop, and Melanie N. Zeilinger. Constrained Inverse Optimal Control With Application to a Human Manipulation Task. *IEEE TCST*, 2021.
- Ali Mesbah. Stochastic model predictive control with active uncertainty learning: A Survey on dual control. *Annual Reviews in Control*, 45:107–117, 2018. ISSN 1367-5788.
- Richard M. Murray, S. Shankar Sastry, and Li Zexiang. *A Mathematical Introduction to Robotic Manipulation*. CRC Press, Inc., USA, 1st edition, 1994. ISBN 0849379814.
- Dimitris Papadimitriou and Daniel S. Brown. Bayesian Constraint Inference from User Demonstrations Based on Margin-Respecting Preference Models. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 15039–15046, 2024.
- Dimitris Papadimitriou and Jingqi Li. Constraint Inference in Control Tasks from Expert Demonstrations via Inverse Optimization. In *2023 62nd IEEE Conference on Decision and Control (CDC)*, pages 1762–1769, 2023.
- Dimitris Papadimitriou, Usman Anwar, and Daniel S. Brown. Bayesian Methods for Constraint Inference in Reinforcement Learning. *Transactions on Machine Learning Research*, 2022.
- Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, MA, 2006.
- Francesco Sabatino. Quadrotor Control: Modeling, Nonlinear Control Design, and Simulation. Ms thesis, KTH, 2015.
- Dorsa Sadigh, S. Shankar Sastry, Sanjit A. Seshia, and Anca Dragan. Information Gathering Actions over Human Internal State. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 66–73, 2016.
- Dorsa Sadigh, Nick Landolfi, Shankar Sastry, Sanjit Seshia, and Anca Dragan. Planning for Cars that Coordinate with People: Leveraging Effects on Human Actions for Planning and Active Information Gathering over Human Internal State. *Autonomous Robots*, 42, 10 2018.
- Sumeet Singh, Jonathan Lacotte, Anirudha Majumdar, and Marco Pavone. Risk-sensitive Inverse Reinforcement Learning via Semi- and Non-parametric Methods. *The International Journal of Robotics Research*, 37(13-14):1713–1740, 2018.

Kaylene C. Stocking, D. Livingston McPherson, Robert P. Matthew, and Claire J. Tomlin. Maximum Likelihood Constraint Inference on Continuous State Spaces. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 8598–8604, 2022.

Andreas Wächter and Lorenz T Biegler. On the Implementation of an Interior-point Filter Line-search Algorithm for Large-scale Nonlinear Programming. *Mathematical Programming*, 106: 25–57, 2006.

James Wilson, Viacheslav Borovitskiy, Alexander Terenin, Peter Mostowsky, and Marc Deisenroth. Efficiently Sampling Functions from Gaussian Process Posteriors. In *International Conference on Machine Learning*, pages 10292–10302. PMLR, 2020.

Appendix A. Supplementary Material for Sec. 2

A.1. Extraction of Tight Points and Constraint Gradient Estimates

Our work uses methods first proposed in Sec. IV-A in Chou et al. (2022) to extract tight system states and compute the associated gradient values. We describe these methods below for completeness.

First, we describe the extraction of *tight* system states from a given locally-optimal demonstration ξ_d . Complementary slackness (2c) encodes that if the constraint $g_{\neg k}(\phi_{\text{sep}}(x_t(\xi_d))) \leq 0$ were *not* tight at some time $t \in [T]$, the corresponding Lagrange multiplier at that time $t \in [T]$ must be zero. In that case, we could enforce the stationarity condition at time $t \in [T]$ while setting the unknown constraint's Lagrange multiplier $\lambda_{d,\neg k}$ equal to $\mathbf{0}$, i.e., $\mathbf{s}_{x_t}(\xi_d, \lambda_{d,k}, \mathbf{0}, \nu_d) = \mathbf{0}$. Conversely, if $g_{\neg k}(\phi_{\text{sep}}(x_t(\xi_d))) = 0$, then $\mathbf{s}_{x_t}(\xi_d, \lambda_{d,k}, \mathbf{0}, \nu_d) \neq \mathbf{0}$ for all possible $\lambda_{d,k}$ and ν_d satisfying dual feasibility (2b) and complementary slackness (2c), and thus the following linear program would yield a strictly positive optimal value (Chou et al. (2022), Corollary 1):

Problem 7 (Tightness Check)

$$\min. \quad \|\mathbf{s}_{x_t}(\xi_d, \lambda_{d,k}, \mathbf{0}, \nu_d)\|_1 \quad (13a)$$

$$\text{s.t.} \quad \lambda_{d,k} \geq \mathbf{0}, \quad \lambda_{d,k} \odot \mathbf{g}_k(\xi_d) = \mathbf{0}. \quad (13b)$$

For each $d \in [D]$, we collect times $t \in [T]$ at which Prob. 7 yields a strictly positive optimal value to construct an *estimated set of tight timesteps* $\mathbf{t}_{\text{tight}}(\xi_d)$, which is a guaranteed under-approximation of the true set of tight timesteps $\{t \in [T] : g_{\neg k}^*(\phi_{\text{sep}}(x_t(\xi_d))) = 0\}$ for the d -th demonstration.

Next, for each demonstration $d \in [D]$, at each $t \in \mathbf{t}_{\text{tight}}(\xi_d)$, we describe a method for generating estimates $w_{d,t}$ for the corresponding gradient of the unknown constraint, i.e., $\nabla_{x_t} g_{\neg k}^*(\phi_{\text{sep}}(x_t(\xi_d)))$. Concretely, following Sec. IV-A in Chou et al. (2022), we first define $\mathbf{1}_{\text{tight}}(\xi_d) \in \mathbb{R}^T$ to be the vector whose t -th component equals 1 if $t \in \mathbf{t}_{\text{tight}}(\xi_d)$, and 0 otherwise, for each $t \in [T]$. Then, we fix $\lambda_{d,\neg k} = \mathbf{1}_{\text{tight}}(\xi_d)$, and solve for gradient estimates $w_{d,t} \in \mathbb{R}^{1 \times n}$ which are compatible with the KKT conditions (2) via the following linear program:

Problem 8 (Constraint Gradient Estimation)

$$\text{find.} \quad \lambda_{d,k}, \nu_d, \{w_{d,t} : t \in \mathbf{t}_{\text{tight}}(\xi_d)\} \quad (14a)$$

$$\text{s.t.} \quad \mathbf{g}_{\neg k}^*(\phi(\xi_d)) \leq \mathbf{0}, \quad \lambda_{d,k} \geq \mathbf{0}, \quad \lambda_{d,k} \odot \mathbf{g}_k(\xi_d) = \mathbf{0}, \quad (14b)$$

$$\nabla_{x_t} c(\xi) + \lambda_k^\top \nabla_{x_t} \mathbf{g}_k^*(\xi) + \mathbb{1}\{t \in \mathbf{t}_{\text{tight}}(\xi_d)\} \cdot w_{d,t} + \nu_k^\top \nabla_{x_t} \mathbf{h}_k(\xi) = \mathbf{0}, \quad \forall t \in [T], \quad (14c)$$

$$\nabla_{u_t} c(\xi) + \lambda_k^\top \nabla_{u_t} \mathbf{g}_k^*(\xi) + \nu_k^\top \nabla_{u_t} \mathbf{h}_k(\xi) = \mathbf{0}, \quad \forall t \in [T]. \quad (14d)$$

Thm. 2 in Chou et al. (2022) provides mild conditions under which $\{w_{d,t} : t \in \mathbf{t}_{\text{tight}}(\xi_d)\}$ recover the true constraint gradients $\{\nabla_{x_t} g_{\neg k}^*(\phi_{\text{sep}}(x_t(\xi_d))) : t \in \mathbf{t}_{\text{tight}}(\xi_d)\}$ up to scale, i.e., for each $t \in \mathbf{t}_{\text{tight}}(\xi_d)$, there exists some $\alpha_{d,t} > 0$ such that $\nabla_{x_t} g_{\neg k}^*(\phi_{\text{sep}}(x_t(\xi_d))) = \alpha_{d,t} w_{d,t}$.

A.2. A Primer on Gaussian Processes and Their Training Process

A Gaussian Process $\mathcal{GP}(m, k)$ is a (possibly infinite) set of random variables characterized by a mean function $m : \mathbb{R}^n \rightarrow \mathbb{R}$ and a covariance function $k : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}$, whose finite sub-collections are always jointly Gaussian (Rasmussen and Williams (2006)). GPs are often employed

as priors in regression tasks, in which one aims to infer an unknown map $g : \mathbb{R}^n \rightarrow \mathbb{R}$ from an input-output dataset $\mathcal{D} := (\{(x_i, y_i) \in \mathbb{R}^n \times \mathbb{R}^m\})_{i=1}^{N_d}$ generated via a noisy output model $y_i \sim \mathcal{N}(g(x_i), \sigma^2)$. In such scenarios, the posterior $(\tilde{g}|\mathcal{D})(\cdot)$ of the function g is characterized by the following posterior mean and covariance maps:

$$\mathbb{E}[\tilde{g}(\cdot)|\mathcal{D}] = k(\cdot, \mathbf{X})(k(\mathbf{X}, \mathbf{X}) + \sigma^2 I)^{-1} \mathbf{Y}, \quad (15a)$$

$$\text{cov}[\tilde{g}(\cdot)|\mathcal{D}] = k(\cdot, \cdot) - k(\cdot, \mathbf{X})(k(\mathbf{X}, \mathbf{X}) + \sigma^2 I)^{-1} k(\mathbf{X}, \cdot) \quad (15b)$$

Whereas [Chou et al. \(2022\)](#) only uses the posterior mean for constraint inference, we additionally sample from the posterior $(\tilde{g}|\mathcal{D})(\cdot)$ to facilitate higher constraint learning efficiency. Concretely, we sample posterior functions $\{\hat{g}_p(\cdot) : p \in [P]\}$ using the Random Fourier Features-based approach presented in [Wilson et al. \(2020\)](#). Concretely, we sample n_ℓ basis functions $\phi_\ell : \mathbb{R}^{n_c} \rightarrow \mathbb{R}$, as described in [Wilson et al. \(2020\)](#), and select a scale coefficient $\alpha > 0$, which modulates the level of random deviations between $\mathbb{E}[\tilde{g}(\cdot)|\mathcal{D}]$ and each posterior function $\hat{g}_p(\cdot)$. We then draw $w_{p,\ell} \sim \text{i.i.d. } \mathcal{N}(0, 1)$, and define, for each $p \in P$:

$$\hat{g}_p(\cdot) := \mathbb{E}[\tilde{g}(\cdot)|\mathcal{D}] + \alpha \cdot \sum_{\ell=1}^{n_\ell} w_{p,\ell} \left(\phi_\ell(\cdot) - k(\cdot, \mathbf{X})(k(\mathbf{X}, \mathbf{X}) + \sigma^2)^{-1} \phi_\ell(\mathbf{X}) \right), \quad (16)$$

to form the desired GP posterior samples.

Appendix B. Supplementary Material for Sec. 4

B.1. Constraint Types

We concretely formulate the constraints $g_{k,1}^*, \dots, g_{k,7}^*$ and associated avoid sets $\mathcal{A}_1, \dots, \mathcal{A}_7$ below. We note that \mathcal{A}_8 is a physical obstacle in a hardware experiment with a 7-DOF robot arm. Recall that, by definition, $\mathcal{A}_i := \{\kappa \in \mathbb{R}^{n_c} : g_{k,i}^*(\kappa) \leq 0\}$ for each $i \in [8]$. As formulated in Sec. 4, given a state x , we refer to the position, x -coordinate position, y -coordinate position, and z -coordinate position by p, p_x, p_y , and p_z , respectively.

For $g_{k,1}^*$ and \mathcal{A}_1 , we set:

$$g_{k,1}^*(\phi_{\text{sep}}(x)) = -0.02|1.732p_x + p_y|^{1.4} - 0.042|p_x - 1.732p_y|^{1.4} + 1 \quad (17)$$

For $g_{k,2}^*$ and \mathcal{A}_2 , we set:

$$g_{k,2}^*(\phi_{\text{sep}}(x)) = -\min_{i \in [2]} \{p^\top Q_i p + q_i^\top p + r_i\}, \quad (18)$$

where:

$$Q_1 = \begin{bmatrix} 0.0868 & 0.0243 \\ 0.0243 & 0.0868 \end{bmatrix}, \quad q_1 = \begin{bmatrix} -0.8403 \\ -0.7153 \end{bmatrix}, \quad R_1 = 1.7534 \quad (19a)$$

$$Q_2 = \begin{bmatrix} 0.0868 & 0.0243 \\ 0.0243 & 0.0868 \end{bmatrix}, \quad q_2 = \begin{bmatrix} 0.8403 \\ 0.7153 \end{bmatrix}, \quad R_2 = 1.7534. \quad (19b)$$

For $g_{k,3}^*$ and \mathcal{A}_3 , we set:

$$g_{k,3}^*(\phi_{\text{sep}}(x)) := -\left(\frac{1}{4}p_x^2 + p_y^2 - 25\right)^2 - 50p_x^2 + 687.5 \quad (20)$$

For $g_{k,4}^*$ and \mathcal{A}_4 , we set:

$$g_{k,4}^*(\phi_{\text{sep}}(x)) := -\max\left\{\frac{1}{5}|p_x|, \frac{1}{4}|p_y|\right\} + 1. \quad (21)$$

For $g_{k,5}^*$ and \mathcal{A}_5 , we set:

$$g_{k,5}^*(\phi_{\text{sep}}(x)) := -\left(\frac{1}{9}x^2 + \frac{1}{9}y^2 + z^2 - 6\right)^2 - \frac{16}{3}(x^2 + y^2) + 39.6. \quad (22)$$

For $g_{k,6}^*$ and \mathcal{A}_6 , we set:

$$g_{k,6}^*(\phi_{\text{sep}}(x)) := -\min_{i \in [3]}(x - v_i)^\top W_i(x - v_i) + R, \quad (23)$$

where:

$$v_1 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \quad W_1 = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad R = 49, \quad (24)$$

$$v_2 = \begin{bmatrix} 5 \\ 0 \\ 5 \end{bmatrix}, \quad W_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 4 \end{bmatrix}, \quad (25)$$

$$v_3 = \begin{bmatrix} -5 \\ 0 \\ -5 \end{bmatrix}, \quad W_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 4 \end{bmatrix}, \quad (26)$$

$$(27)$$

For $g_{k,7}^*$ and \mathcal{A}_7 , we set:

$$g_{k,7}^*(\phi_{\text{sep}}(x)) := -x^2 B^{-2} x + 1, \quad (28)$$

where $B := \text{diag}\{150, 90, 150, 90, 150, 90, 150\} \in \mathbb{R}^{7 \times 7}$. Here, $\text{diag}\{\cdot\}$ describes a diagonal matrix, with the given entries indicating diagonal values.

B.2. Additional Experiment Results

2D Double Integrator Experiments We evaluate our GP-ACL algorithm on the problem of recovering the three nonlinear constraints $g_{k,1}^*$, $g_{k,2}^*$, $g_{k,3}^*$, and $g_{k,4}^*$, as defined in B.1 and visualized in Figs. 4, 5, 6, and 7. Here, we set $\alpha = 0.3$ when sampling GP posteriors for learning $g_{k,3}^*$, and use the default setting of $\alpha = 0.15$ while learning $g_{k,1}^*$, $g_{k,2}^*$, $g_{k,3}^*$, and $g_{k,4}^*$. All other parameters are set at the default values provided in Sec. 4.1. Across $n_s = 2500$ sampled constraint states, our GP-ACL algorithm accurately predicts the safeness or unsafeness of each sample with accuracy $\gamma_{\text{ours}} = 0.9952, 0.996, 0.9956, \text{ and } 0.9762$ for the constraints $g_{k,1}^*$, $g_{k,2}^*$, $g_{k,3}^*$, and $g_{k,4}^*$, respectively, while the random sampling-based baseline method yielded accuracies of only $\gamma_{\text{BL}} = 0.99, 0.9868, 0.9652, \text{ and } 0.9692$, respectively.

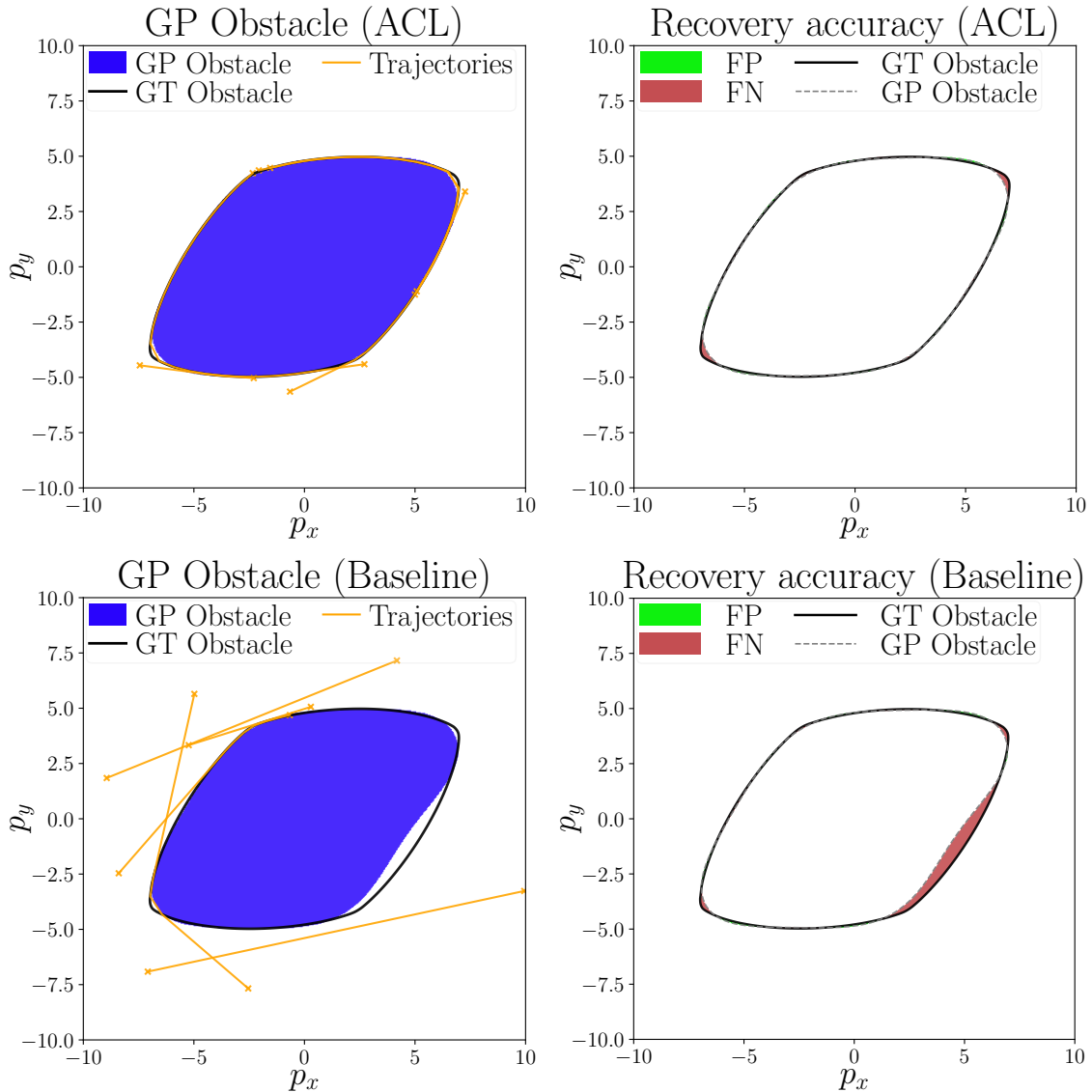


Figure 4: Our GP-ACL algorithm (top) outperforms the random sampling baseline (bottom) in accurately recovering the nonlinear constraint $g_{\gamma_{k,1}}^*$ from demonstrations generated using 2D double integrator dynamics, with fewer false positive (green) and fewer false negative (red) errors.

3D Double Integrator Experiments We further evaluate our GP-ACL algorithm by recovering the complex nonlinear constraint $g_{\gamma_{k,6}}^*$, as defined in B.1 and visualized in Fig. 8, using demonstrations of length $T = 20$. Across $n_s = 2500$ sampled constraint states, our GP-ACL algorithm accurately predicts the safeness or unsafeness of each sample with accuracy $\gamma_{\text{ours}} = 0.9914$, while the random sampling-based baseline method yielded accuracies of only $\gamma_{\text{BL}} = 0.9724$.

Unicycle Experiments (Additional) We further evaluate our GP-ACL algorithm by recovering the constraint $g_{\gamma_{k,4}}^*$, as defined in B.1 and visualized in Fig. 9, using demonstrations of length

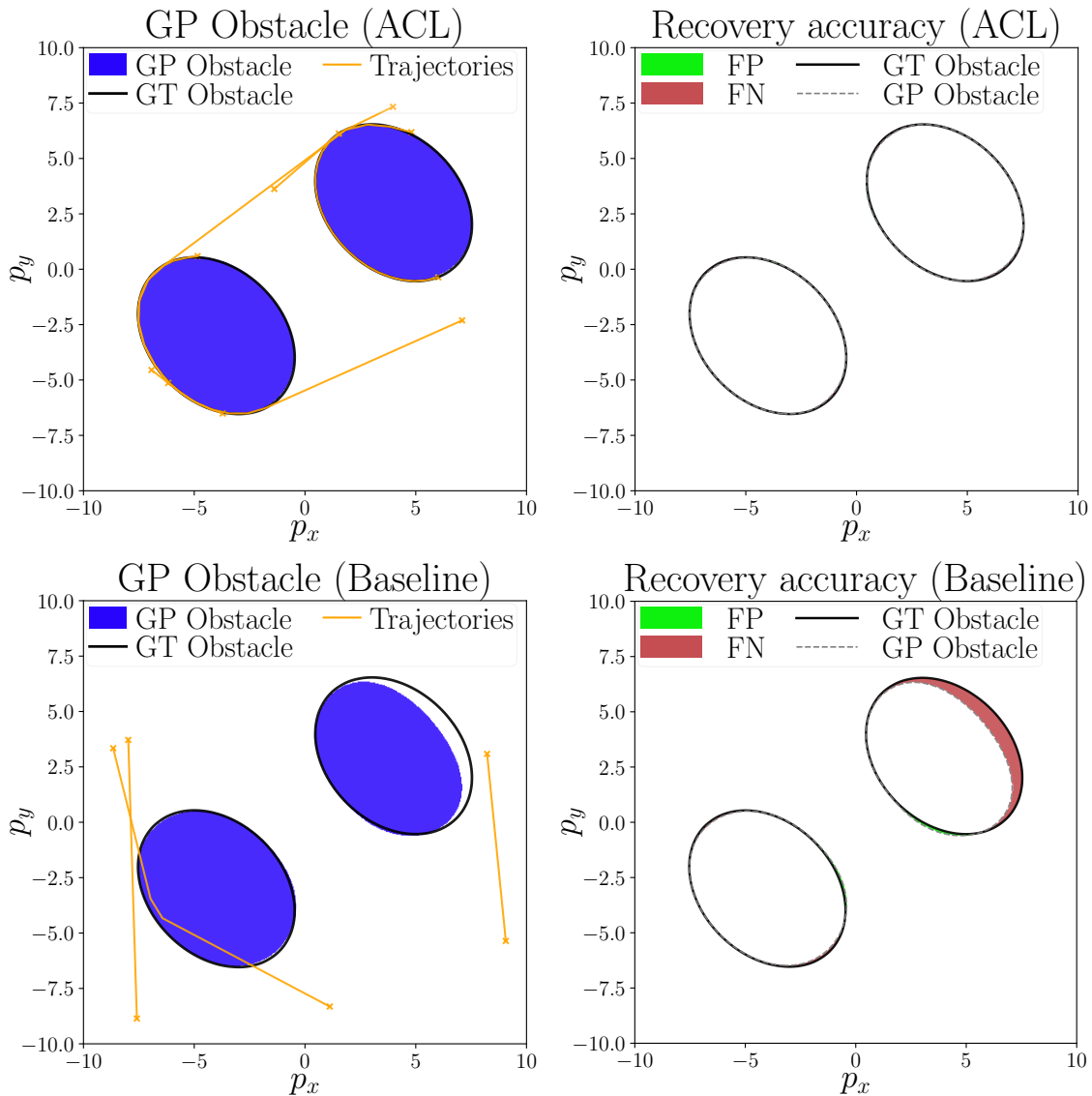


Figure 5: Our GP-ACL algorithm (top) outperforms the random sampling baseline (bottom) in accurately recovering the nonlinear constraint $g_{\tau_{k,2}}^*$ from demonstrations generated using 2D double integrator dynamics, with fewer false positive (green) and fewer false negative (red) errors.

$T = 30$. As shown in Fig. 9, our GP-ACL algorithm predicts the safeness or unsafeness of each sample more accurately than the baseline method.

7-DOF Arm Hardware Experiments We apply our GP-ACL algorithm on a physical 7-DOF Kuka LBR iiwa 7 to learn the “C-shaped” constraint (see Fig. 1), described mathematically as a planar set $\mathcal{A}_8 \subseteq \mathbb{R}^2$ within the robot arm’s workspace. Demonstrations are of length $T = 20$, and are collected via end effector teleoperation.

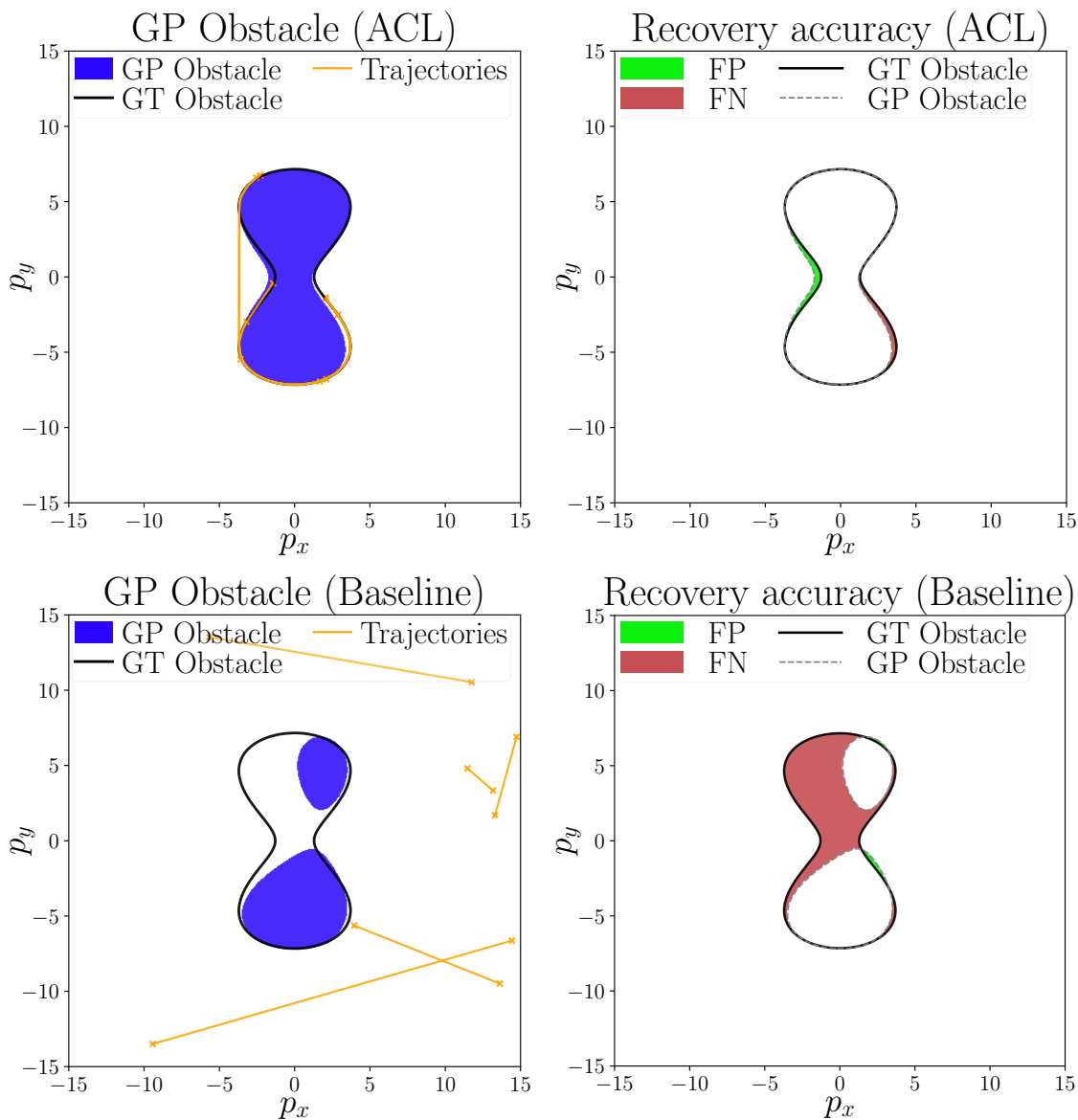


Figure 6: Our GP-ACL algorithm (top) outperforms the random sampling baseline (bottom) in accurately recovering the nonlinear constraint $g_{7,k,3}^*$ from demonstrations generated using 2D double integrator dynamics, with fewer false positive (green) and fewer false negative (red) errors.

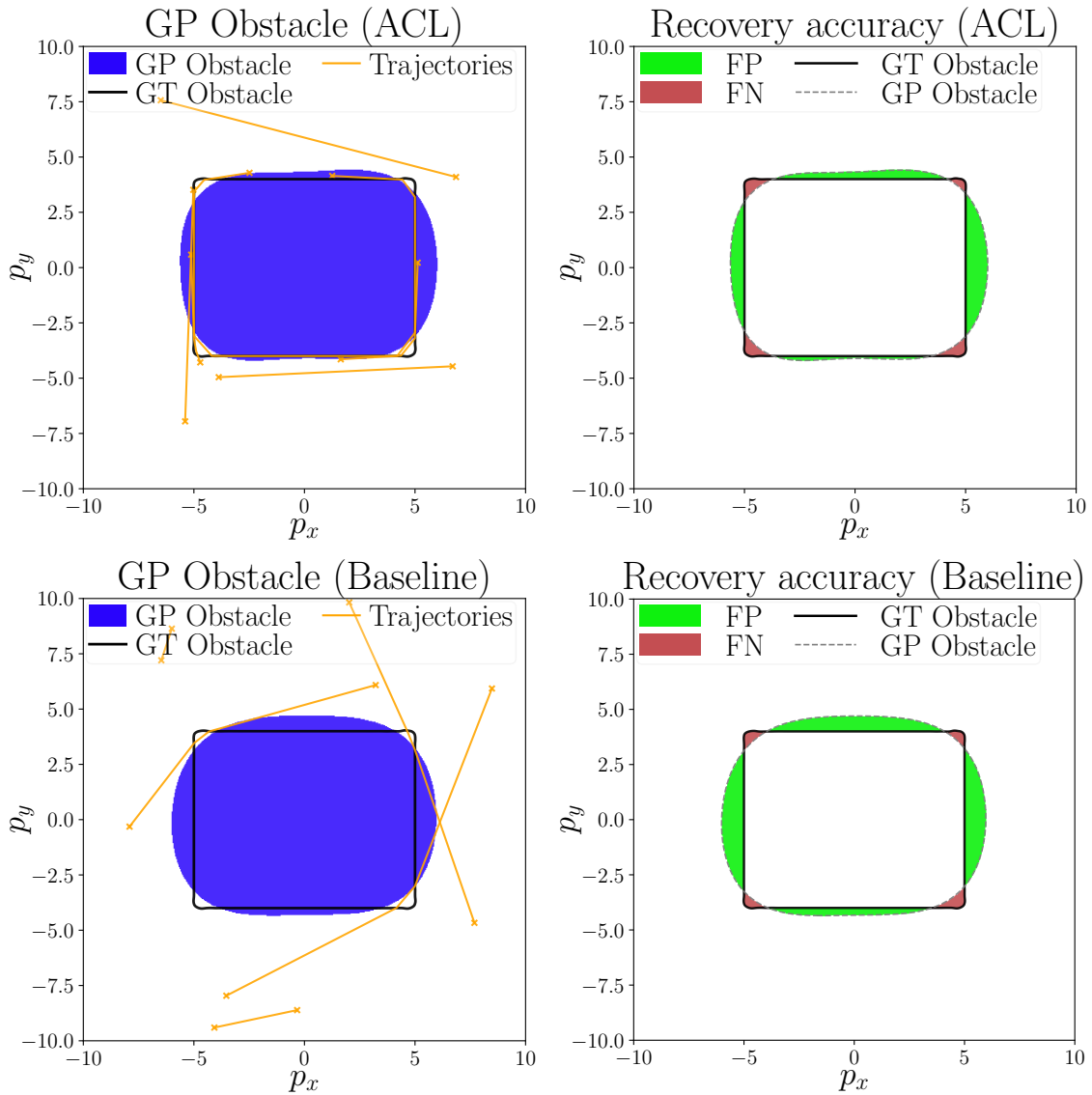


Figure 7: Our GP-ACL algorithm (top) outperforms the random sampling baseline (bottom) in accurately recovering the nonlinear constraint $g_{\tau_{k,4}}^*$ from demonstrations generated using 2D double integrator dynamics, with fewer false positive (green) and fewer false negative (red) errors.

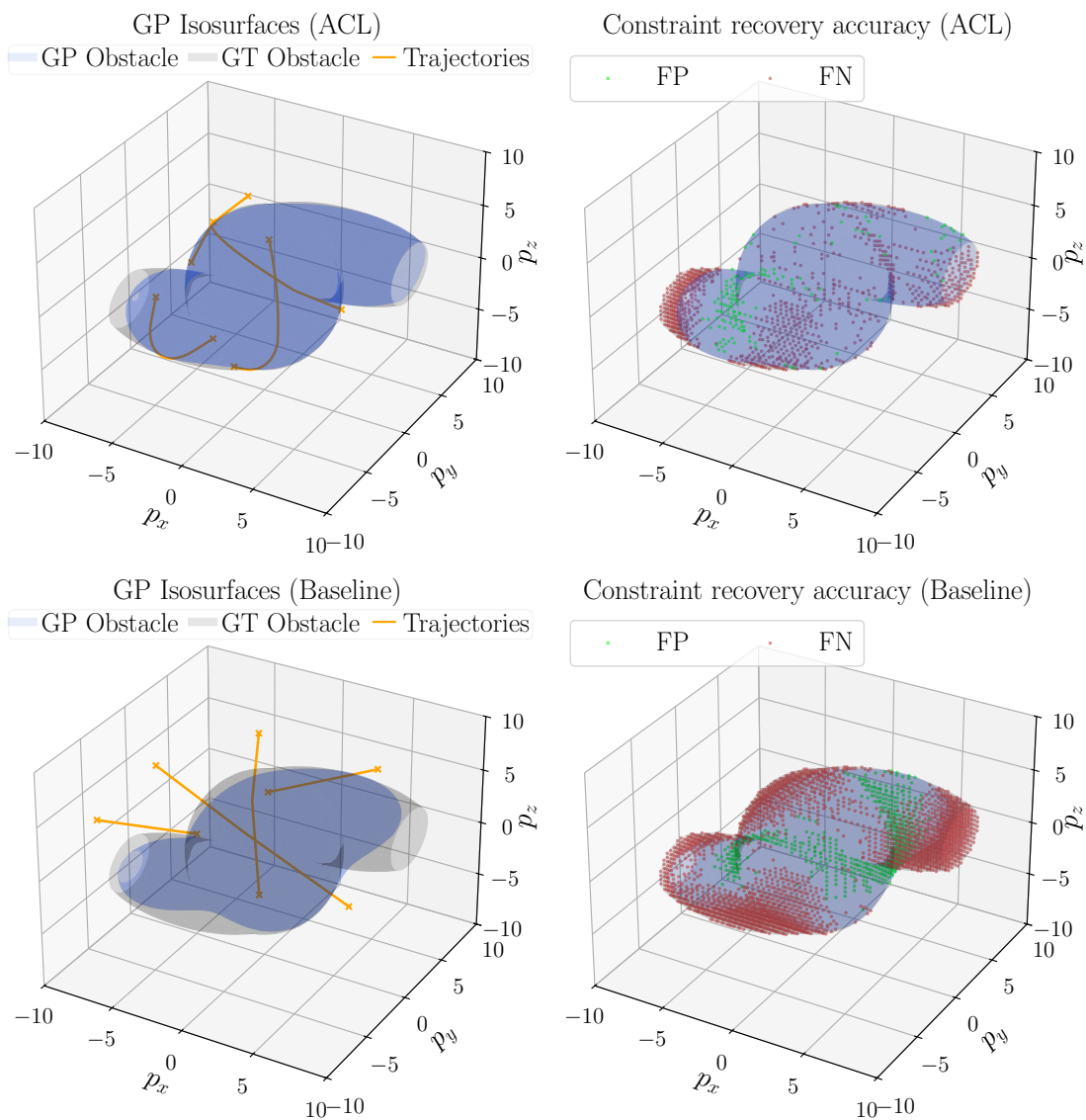


Figure 8: Our GP-ACL algorithm (top) outperforms the random sampling baseline (bottom) in accurately recovering the nonlinear constraint $g_{7k,6}^*$ from demonstrations generated using 3D double integrator dynamics, with fewer false positive (green) and false negative (red) errors.

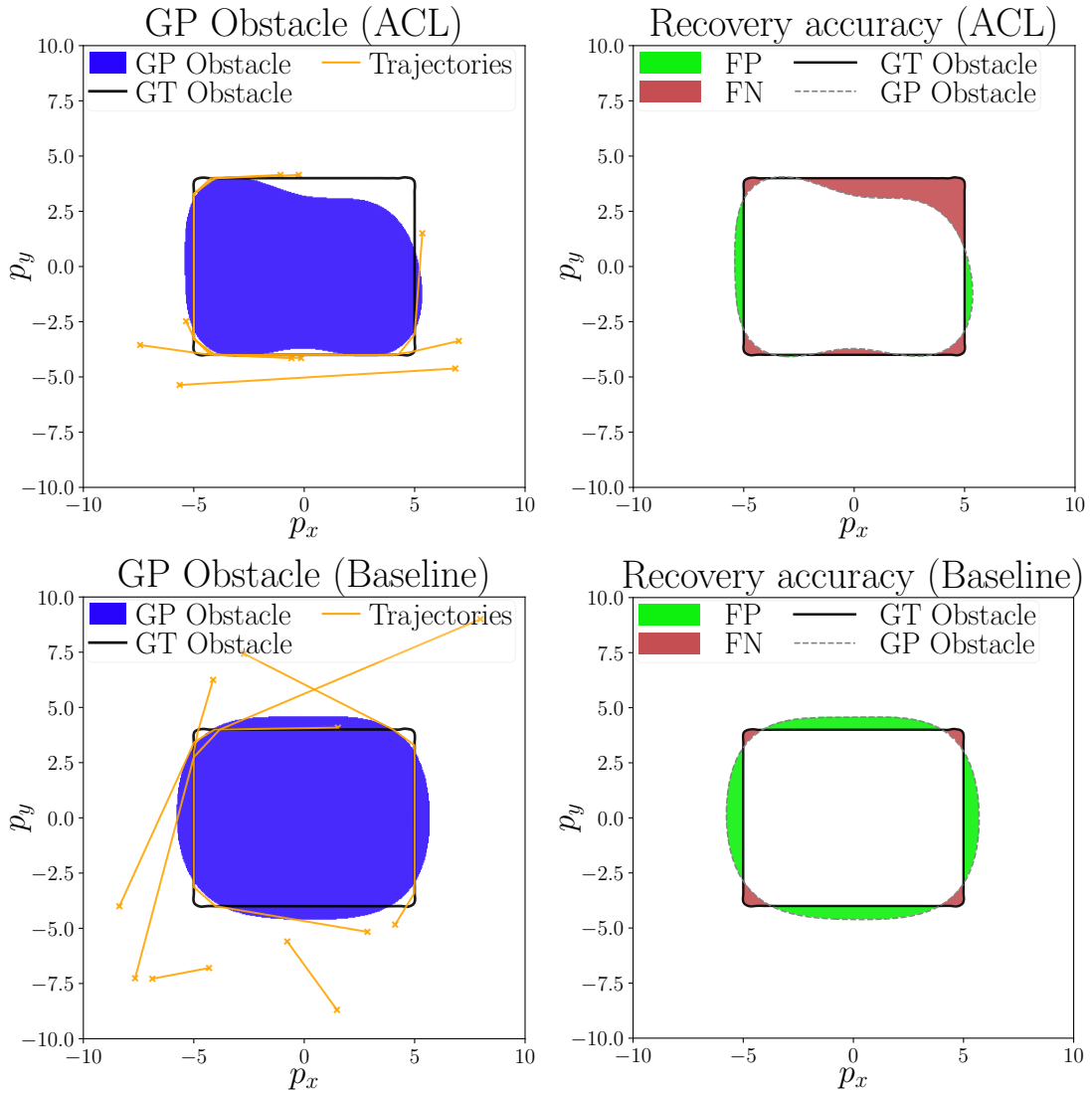


Figure 9: Our GP-ACL algorithm (top) outperforms the random sampling baseline (bottom) in accurately recovering the constraint $g_{\nabla k,4}^*$ from demonstrations generated using 4D unicycle dynamics, with fewer false positive (green) and false negative (red) errors.

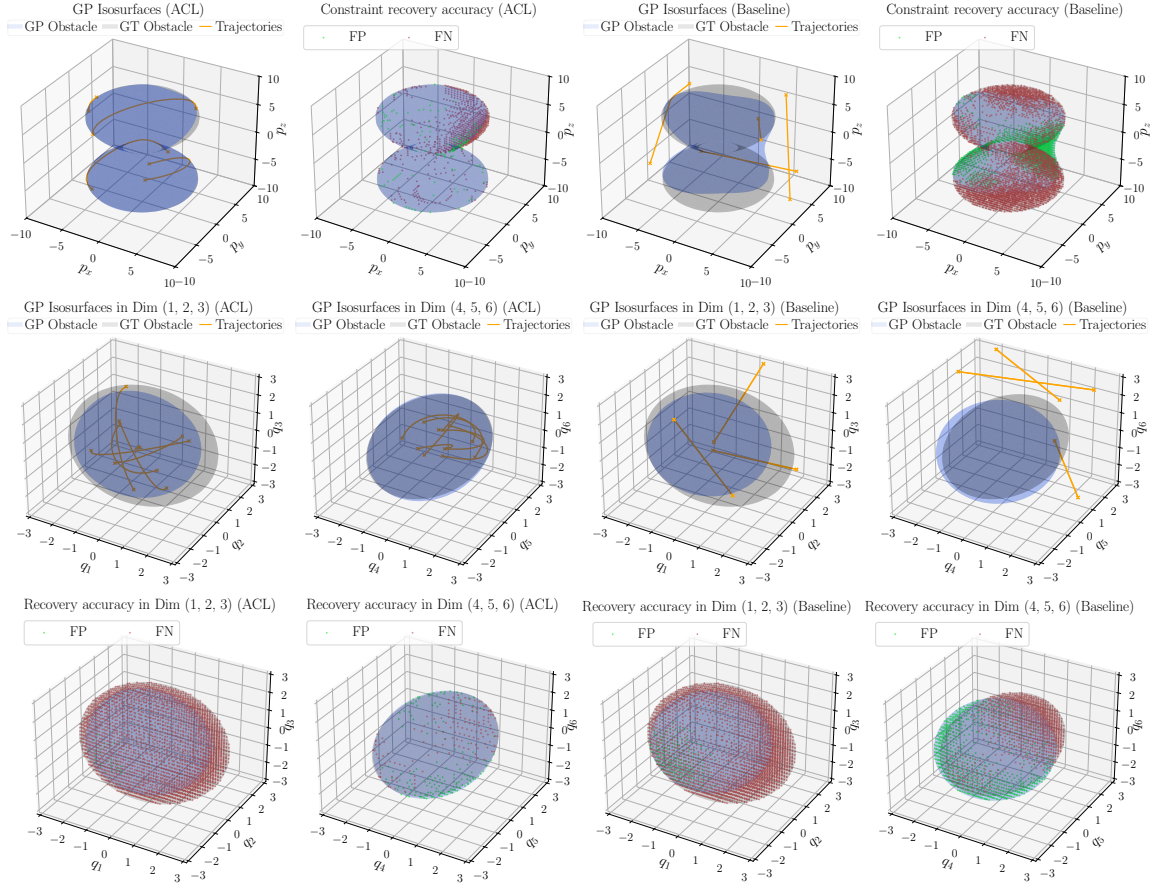


Figure 10: Our GP-ACL algorithm (left half) outperforms the random sampling baseline (right half) in accurately recovering constraints $g_{k,3}^*$ (top), $g_{k,7}^*$ (middle/bottom), from unicycle dynamics (top) and simulated 7-DOF arm (middle/bottom) demonstrations, with fewer false positive (green) and false negative (red) errors. Middle and bottom row figures display 3D slices of the 7D constraint space from our numerical simulations on the 7-DOF arm.

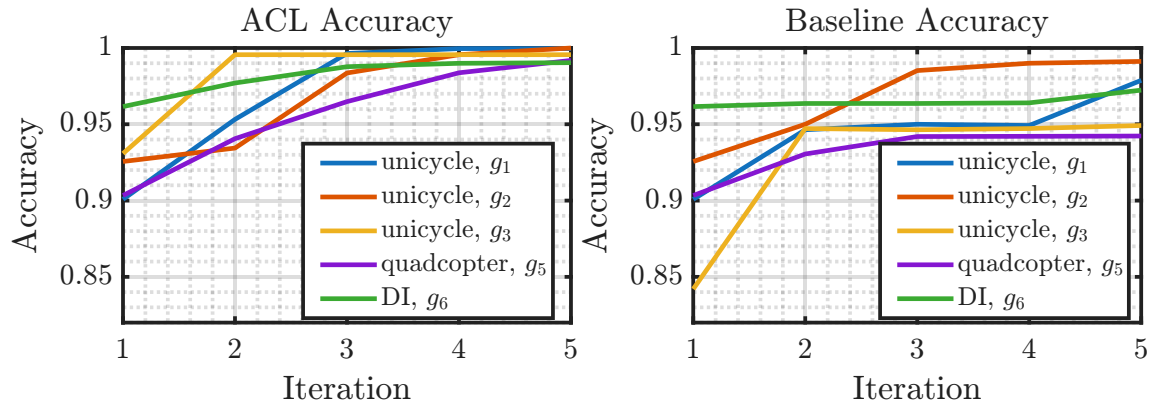


Figure 11: Constraint recovery accuracy of (a) our GP-ACL algorithm and (b) the random-sampling baseline approach. Across the following constraint learning tasks, our GP-ACL method consistently recovered the *a priori* unknown constraints more accurately compared to the random-sampling baseline method: Learning the constraints $g_{k,1}^*$, $g_{k,2}^*$, and $g_{k,3}^*$ from demonstrations generated using unicycle dynamics (blue, red, and yellow, respectively); learning the constraint $g_{k,5}^*$ from quadcopter dynamics (purple); and learning the constraint $g_{k,6}^*$ from double integrator dynamics (green).

