Uncertainty-Aware Constraint Learning for Adaptive Safe Motion Planning from Demonstrations

Glen Chou, Necmiye Ozay, and Dmitry Berenson
Department of Electrical Engineering and Computer Science
University of Michigan, Ann Arbor
{gchou, necmiye, dmitryb}@umich.edu

Abstract: We present a method for learning to satisfy uncertain constraints from demonstrations. Our method uses robust optimization to obtain a belief over the potentially infinite set of possible constraints consistent with the demonstrations, and then uses this belief to plan trajectories that trade off performance with satisfying the possible constraints. We use these trajectories in a closed-loop policy that executes and replans using belief updates, which incorporate data gathered during execution. We derive guarantees on the accuracy of our constraint belief and probabilistic guarantees on plan safety. We present results on a 7-DOF arm and 12D quadrotor, showing our method can learn to satisfy high-dimensional (up to 30D) uncertain constraints, and outperforms baselines in safety and efficiency.

Keywords: learning from demonstration, planning under uncertainty, safe learning

1 Introduction

Learning from demonstration (LfD) is a powerful framework for teaching robots to perform tasks. In particular, recent work has shown that modeling tasks as constrained optimizations, and learning the cost function and constraints from demonstrations [1, 2, 3, 4], can enable the learning of complex manipulation and mobile robotics tasks. However, a core problem in LfD, and constraint-learning in particular, is the unidentifiability of the constraints: there is often an infinite set of *possible constraints* which are sufficient to explain a demonstration. While previous work [1] has evaded this problem when planning with the learned constraint by planning guaranteed-safe trajectories that satisfy *all possible constraints* consistent with the data, this is impossible in most realistic scenarios, where the set of possible constraints is so large that the planning problem becomes infeasible. For example, consider planning for an arm in a cluttered home environment: unless the demonstrations activate each of the multitude of constraints, we cannot claim that a trajectory is guaranteed-safe.

Our insight is that to plan under large constraint uncertainty, it is vital to plan trajectories that trade off safety and efficiency by reasoning over the set of possible constraints, and to update this set using constraint information gathered when executing these trajectories. Specifically, we leverage robust optimization and Bayesian inference to obtain and update our belief over the possible constraints consistent with the demonstrations and gathered data. Then, we propose a policy for adaptively satisfying the constraints which interleaves chance-constrained planning, execution, and belief updates until the task is completed. This paper makes following specific contributions:

1. We show how to extract *all possible constraints*, for some constraint parameterization, consistent with a set of locally-optimal demonstrations, which we use to construct a belief over constraints.

2. We provide a novel method for planning approximately-optimal open-loop trajectories between new start and goal states, which are safe under the constraint belief with a prescribed probability.

3. We show how to use these open-loop trajectories to construct a closed-loop policy to adaptively satisfy the uncertain constraints, incorporating constraint information observed during execution.

4. We theoretically analyze our algorithm, proving the completeness of constraint extraction for various constraint parameterizations and providing probabilistic safety guarantees for our planner.

5. We evaluate our method by planning for a 7-DOF arm and a quadrotor with uncertain high-dimensional constraints, showing that our methods outperform baselines in efficiency and safety.

2 Related Work

The constraint learning literature has shown how to learn global constraints shared across demonstrations [1, 3, 5], geometric state space constraints [6, 7], and local trajectory constraints [8, 9, 10, 11]. However, prior work makes the key assumption that the demonstrations are sufficiently informative for the learned constraint to be confidently used to plan new trajectories. As constraint learning is an ill-posed inverse problem, this is often untrue, and the robot should instead reason over the constraint uncertainty to complete the task. Two exceptions are [12], which learns to satisfy linear constraints in iterative tasks – whereas we learn to satisfy non-convex constraints to complete a task once – and [13], which plans open-loop to satisfy a finite set of possible temporal logic constraints – while we plan closed-loop to adaptively satisfy an infinite set of possible low-level state space constraints.

Our work also relates to inverse optimal control [14, 15, 16], imitation learning [17], and safety-focused variants [18, 19, 20, 21, 22, 23] that estimate the uncertainty in the learned policy. However, these approaches only use the uncertainty as a sign to switch to a safe backup policy [18, 20, 23], to evaluate the quality of a given policy [19, 22], or to query the demonstrator for more data [21]. In our setting, we must use the uncertainty from the demonstrations to compute the policy. Further, as we model the demonstrator's preferences and the task constraints separately, we can specifically reason over the uncertain constraints that can be learned in a self-supervised way, without the demonstrator.

Finally, our work relates to planning in uncertain environments. Some methods simplify the problem using Gaussian uncertainty [24] or constraint convexity [25]; however, our constraint belief cannot be represented with these approximations. Other methods sample possible constraints and enforce them all, but this leads to plan infeasibility [26], while [27, 28] focus on high-quality short-range planning due to minimal knowledge of the global map. Our contributions are to show how demonstrations can reduce environment uncertainty, yielding better global plans, and to provide a method to plan chance-constrained trajectories for a class of uncertainty distributions with complex support.

3 Preliminaries and Problem Setup

We consider demonstrations performed on systems $x_{t+1} = f(x_t, u_t, t)$, $x \in \mathcal{X}$, $u \in \mathcal{U}$ completing tasks $\sigma \in \Sigma$, represented as constrained optimizations over state/control trajectories $\xi_{xu} \doteq (\xi_x, \xi_u)$:

Problem 1 (Forward (demonstrator's) problem / "task" σ)**.**

$$
\begin{aligned}
&\underset{\xi_{xu}}{\text{minimize}} && c_\sigma(\xi_{xu}) \\
&\text{subject to} && \phi(\xi_{xu}) \in \mathcal{S}(\theta) \subseteq \mathcal{C} && \Leftrightarrow && \mathbf{g}_{\neg k}(\xi_{xu}, \theta) \leq \mathbf{0} \\
& && \bar{\phi}(\xi_{xu}) \in \bar{\mathcal{S}} \subseteq \bar{\mathcal{C}}, \quad \phi_\sigma(\xi_{xu}) \in \mathcal{S}_\sigma \subseteq \mathcal{C}_\sigma && \Leftrightarrow && \mathbf{h}_k(\xi_{xu}) = \mathbf{0}, \quad \mathbf{g}_k(\xi_{xu}) \leq \mathbf{0}
\end{aligned}
$$

where $c_\sigma(\cdot)$ is task-dependent and $\phi(\cdot)$ maps from trajectories to constraint space $\mathcal{C}$; elements of $\mathcal{C}$ are denoted *constraint states* $\kappa \in \mathcal{C}$. $\bar{\phi}(\cdot)$ and $\phi_\sigma(\cdot)$ map to constraint spaces $\bar{\mathcal{C}}$ and $\mathcal{C}_\sigma$, containing a known shared safe set $\bar{\mathcal{S}}$ and task-dependent safe set $\mathcal{S}_\sigma$; we embed the dynamics in $\bar{\mathcal{S}}$ and start/goal constraints in $\mathcal{S}_\sigma$. We group the constraints of Prob. 1 as (in)equality (ineq/eq) and (un)known ($\neg k/k$), where $\mathbf{h}_k(\xi_{xu}) \in \mathbb{R}^{N_k^{\text{eq}}}$, $\mathbf{g}_k(\xi_{xu}) \in \mathbb{R}^{N_k^{\text{ineq}}}$, and $\mathbf{g}_{\neg k}(\xi_{xu}, \theta) \in \mathbb{R}^{N_{\neg k}^{\text{ineq}}}$, and let $g(\kappa, \theta) \doteq \max_{i \in \{1, \ldots, N_{\neg k}^{\text{ineq}}\}}(g_{i, \neg k}(\kappa, \theta))$. Let the unknown safe and unsafe sets defined by parameter $\theta \in \Theta \subseteq \mathbb{R}^d$ be $\mathcal{S}(\theta)$ and $\mathcal{A}(\theta)$, respectively:

$$
\mathcal{S}(\theta) \doteq \{\kappa \in \mathcal{C} \mid g(\kappa, \theta) \leq 0\} \tag{1}
$$

$$
\mathcal{A}(\theta) \doteq \mathcal{S}(\theta)^c = \{\kappa \in \mathcal{C} \mid g(\kappa, \theta) > 0\} \tag{2}
$$

We assume each state-control demonstration ξ^{loc} approximately solves Prob. 1 to local optimality, satisfying Prob. 1's Karush-Kuhn-Tucker (KKT) conditions [29] within a tolerance. Intuitively, this means ξ^{loc} is feasible for Prob. 1 (it remains within the safe set $\mathcal{S}(\theta)$ and satisfies the known constraints) and is within the neighborhood of a local optimum. With Lagrange multipliers λ, ν, the relevant KKT conditions for the jth demonstration ξ_j^{loc}, denoted $\text{KKT}(\xi_j^{\text{loc}})$, are:

$$\mathbf{g}_{\neg k}(\xi_j^{\text{loc}}, \theta) \leq \mathbf{0} \tag{3a}$$

$$\boldsymbol{\lambda}_k^j \odot \mathbf{g}_k(\xi_j^{\text{loc}}) = \mathbf{0}, \qquad \boldsymbol{\lambda}_k^j \geq \mathbf{0} \tag{3b}$$

$$\boldsymbol{\lambda}_{\neg k}^j \odot \mathbf{g}_{\neg k}(\xi_j^{\text{loc}}, \theta) = \mathbf{0}, \qquad \boldsymbol{\lambda}_{\neg k}^j \geq \mathbf{0} \tag{3c}$$

$$\nabla_{\xi_{xu}} c_\sigma(\xi_j^{\text{loc}}) + \boldsymbol{\lambda}_k^{j\top} \nabla_{\xi_{xu}} \mathbf{g}_k(\xi_j^{\text{loc}}) + \boldsymbol{\lambda}_{\neg k}^{j\top} \nabla_{\xi_{xu}} \mathbf{g}_{\neg k}(\xi_j^{\text{loc}}, \theta) + \boldsymbol{\nu}_k^{j\top} \nabla_{\xi_{xu}} \mathbf{h}_k(\xi_j^{\text{loc}}) = \mathbf{0} \tag{3d}$$

where $\nabla_{\xi_{xu}}(\cdot)$ differentiates with respect to a flattened ξ_{xu} and $\odot$ denotes elementwise product. (3a) enforces primal feasibility, (3b)-(3c) enforces complementary slackness, and stationarity (3d) enforces the demonstration cannot be

Problem 2 (Inverse constraint learning problem).

$$\begin{aligned} \text{find} \quad & \theta, \mathcal{L} \doteq \{\boldsymbol{\lambda}_k^j, \boldsymbol{\lambda}_{\neg k}^j, \boldsymbol{\nu}_k^j\}_{j=1}^{N_{\text{dem}}} \\ \text{subject to} \quad & \{\text{KKT}(\xi_j^{\text{loc}})\}_{j=1}^{N_{\text{dem}}} \end{aligned}$$

locally improved. As in [1], we can solve Prob. 2 to find constraints that make the demonstrations locally-optimal by finding a θ and Lagrange multipliers which are together consistent with the KKT conditions of the demonstrations. To handle approximate local-optimality in Prob. 2, we relax constraints (3b)-(3d) to penalties. This framework can also learn unknown cost function parameters (c.f. App. B). Let $\mathcal{F}$ denote the feasible set of Prob. 2. Denote the projection of $\mathcal{F}$ onto Θ (the set of all consistent constraints θ) as $\mathcal{F}_\theta$, and the projection of $\mathcal{F}_\theta$ onto $\mathcal{C}$ (the set of possibly-unsafe constraint states) as $\text{proj}_{\mathcal{C}}(\mathcal{F}_\theta)$:

$$\mathcal{F}_\theta \doteq \{\theta \mid \exists \mathcal{L} : (\theta, \mathcal{L}) \in \mathcal{F}\} \tag{4}$$

$$\text{proj}_{\mathcal{C}}(\mathcal{F}_\theta) \doteq \{\kappa \in \mathcal{C} \mid \exists \theta \in \mathcal{F}_\theta, g(\kappa, \theta) > 0\} \tag{5}$$

While Prob. 2 returns *one possible* θ, there can be an *infinite set* of possible θ: $\mathcal{F}_\theta$. Thus, $\mathcal{F}_\theta$ represents the constraint uncertainty. We use $\mathcal{F}_\theta$ to build a constraint belief (Sec. 4) and use $\text{proj}_{\mathcal{C}}(\mathcal{F}_\theta)$ for planning (Sec. 5). Finally, let the set of learned guaranteed-safe/unsafe constraint states be $\mathcal{G}_s$ and $\mathcal{G}_{\neg s}$, where κ is learned guaranteed (un)safe if it is marked (un)safe for all $\theta \in \mathcal{F}_\theta$ (c.f. Fig. 2):

$$\mathcal{G}_s \doteq \bigcap_{\theta \in \mathcal{F}_\theta} \{\kappa \mid g(\kappa, \theta) \leq 0\} \tag{6}$$

$$\mathcal{G}_{\neg s} \doteq \bigcap_{\theta \in \mathcal{F}_\theta} \{\kappa \mid g(\kappa, \theta) > 0\} \tag{7}$$

Previous work [1] plans guaranteed-safe trajectories by enforcing that they always remain in $\mathcal{G}_s$, but $\mathcal{G}_s$ can be tiny or disconnected (Fig. 2), making planning infeasible. In this work, we allow plans to pass through possibly-unsafe space $\text{proj}_{\mathcal{C}}(\mathcal{F}_\theta)$, but seek to minimize constraint violations.

Problem statement: We are given N_{dem} demonstrations $\{\xi_j^{\text{loc}}\}_{j=1}^{N_{\text{dem}}}$, known shared and task-dependent safe sets $\bar{\mathcal{S}} / \mathcal{S}_\sigma$, and a prior $p(\theta)$ over the unknown constraint. Our goals are: 1) recover the set of all constraint parameters $\mathcal{F}_\theta \subseteq \Theta$ consistent with the demonstrations to obtain a constraint belief $b_{\text{dem}}(\theta) \doteq p(\theta \mid \{\xi_j^{\text{loc}}\}_{j=1}^{N_{\text{dem}}}) \in \mathcal{P}(\Theta)$, and 2) compute a policy $\pi(\cdot, \cdot, \cdot) : \mathcal{P}(\Theta) \times \mathcal{X} \times (\mathcal{O})^* \to \mathcal{U}$, which takes a prior, start state x_0, and a sequence of constraint observations, and returns a control input u, and completes a task (in this paper, we consider a task as reaching a goal x_g from x_0) while minimizing one of two objectives. In the first variant, denoted Prob. MCV, we want to reach the goal with the <u>m</u>inimum number of expected <u>c</u>onstraint <u>v</u>iolations. In the second variant, denoted Prob. MEC, we want to <u>m</u>inimize the <u>e</u>xpected <u>c</u>ost of some general objective function.

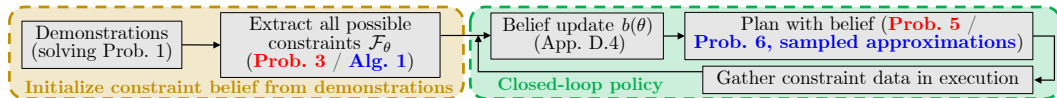| Initialize constraint belief from demonstrations		Closed-loop policy	
Demonstrations (solving Prob. 1)	Extract all possible constraints $\mathcal{F}_\theta$ (**Prob. 3 / Alg. 1**)	Belief update $b(\theta)$ (App. D.4)	Plan with belief (**Prob. 5 / Prob. 6, sampled approximations**)
			Gather constraint data in execution

Fig. 1: Overall method flow for adaptive planning from demonstrations. We refer to both the ideal (red), but intractable, subproblems, as well as the tractable (blue) variants of those subproblems.

Method overview: To prime the reader, we outline two variants of our method in Fig. 1: 1) an ideal variant that requires the solution of intractable optimizations, and 2) tractable variants which approximate the idealized problems or exploit simplifying problem structure. For closed-loop planning, both variants compute the constraint belief (Sec. 4), then iteratively plan with the belief, update the belief with constraint data measured in execution, and replan with the updated belief (Sec. 5).

4 Obtaining a belief over constraints

Extracting $\mathcal{F}_\theta$ is crucial for obtaining an accurate belief over constraints. In this section, we show how to use robust optimization to obtain $\mathcal{F}_\theta$ for some constraint parameterizations. We can robustify Prob. 2, where θ is considered as an uncertain variable in uncertainty set $\hat{\mathcal{F}}_\theta \subseteq \Theta$:

Problem 3 (Inverse constraint learning, robustified in θ).

$$\sup_{\hat{\mathcal{F}}_\theta} \quad \text{Vol}(\hat{\mathcal{F}}_\theta)$$
$$\text{s.t.} \quad \forall \theta \in \hat{\mathcal{F}}_\theta, \ \exists \{\boldsymbol{\lambda}_k^j, \boldsymbol{\lambda}_{\neg k}^j, \boldsymbol{\nu}_k^j \mid \text{KKT}(\xi_j^{\text{loc}})\}_{j=1}^{N_{\text{dem}}}$$

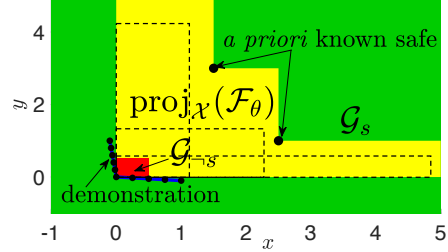

Fig. 2: $\mathcal{F}_\theta$ for a one-box parameterization of $\mathcal{A}(\theta)$, induced by a demo. and two safe states, projected onto $\mathcal{X}$. With the data, the upper x / y bounds $\bar{x}(\theta)$ / $\bar{y}(\theta)$ remain uncertain. Also: some possible $\mathcal{A}(\theta)$ (dotted).
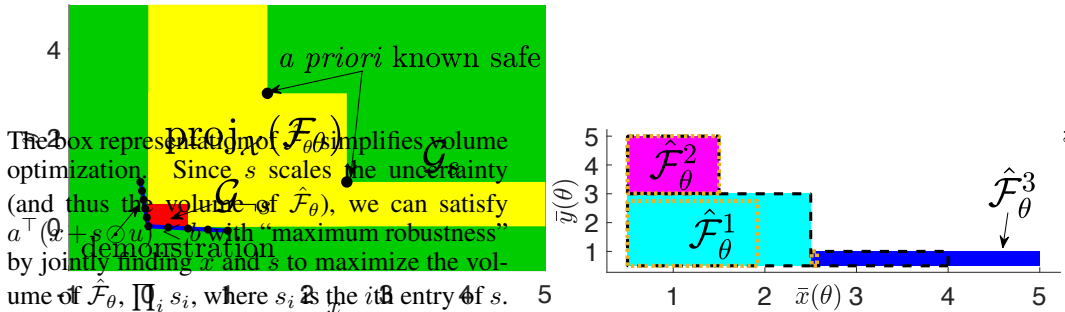
and search for the largest set $\hat{\mathcal{F}}_\theta \subseteq \Theta$ where each $\theta \in \hat{\mathcal{F}}_\theta$ satisfies KKT; the optimizer of Prob. 3 is $\mathcal{F}_\theta$. However, Prob. 3 is intractable due to 1) the optimization over arbitrarily-shaped sets $\hat{\mathcal{F}}_\theta$, 2) measuring the volume of such sets, and 3) the existential quantifiers $\exists$, implying we may need to find different Lagrange multipliers for each $\theta \in \hat{\mathcal{F}}_\theta$. We address these challenges in the following.

4.1 Obtaining the set of demonstration-consistent constraints $\mathcal{F}_\theta$

We assume the unknown constraint $\mathcal{A}(\theta)$ can be represented as a union of boxes in constraint space $\{\kappa \mid \bigcup_i [I, -I]^\top \kappa \leq \theta_i\}$. This assumption is reasonable as any shape can be represented by unioning enough boxes [30], though this can be inefficient (thus, we relax the assumption in App. C.1). In App. B.3.1, we prove that if $\mathcal{A}(\theta)$ can be described as a union of boxes, so can $\mathcal{F}_\theta$. By exploiting this structure, we develop a tractable variant of Prob. 3 using robust linear programming [31].

We address the challenge of set optimization by optimizing over only boxes. Using the identity $\sup_{\|u\|_\infty \leq 1} a^\top u = \|a\|_1$, a linear constraint $a^\top(x + s \odot u) \leq b$ involving uncertain variable u : $\|u\|_\infty \leq 1$, can be equivalently written without u as $a^\top x + \|a \odot s\|_1 \leq b$, where $s \in \mathbb{R}_{\geq 0}^d$ scales the uncertainty. We can use this idea to enforce that the KKT conditions robustly hold everywhere in some box $\theta + s \odot u$, where $\|u\|_\infty \leq 1$. Concretely, we can replace (3a) with $\mathbf{g}_{\neg k}(\xi_j^{\text{loc}}, \theta + s \odot u) \leq$ **0**, (3c) with $\boldsymbol{\lambda}_{\neg k}^j \odot \mathbf{g}_{\neg k}(\xi_j^{\text{loc}}, \theta + s \odot u) = \mathbf{0}$, and (3d) with $\nabla_\xi \ c(\xi_j^{\text{loc}}) + \boldsymbol{\lambda}_k^{j\top} \nabla_\xi \ \mathbf{g}_k(\xi_j^{\text{loc}}) +$

The box representation simplifies volume optimization. Since s scales the uncertainty (and thus the volume of $\hat{\mathcal{F}}_\theta$), we can satisfy $a^\top(x + s \odot u) \leq b$ with "maximum robustness" by jointly finding x and s to maximize the volume of $\hat{\mathcal{F}}_\theta$, $\prod_i s_i$, where s_i is the ith entry of s. While $\prod_i s_i$ is non-convex in s, its geometric mean, $(\prod_i s_i)^{1/d}$, is conic-representable [32]. It is also a monotonic transform of the volume, and thus an exact surrogate for volume maximization.

Fig. 3: Extracting $\mathcal{F}_\theta$ via Alg. 1: requires 3 iterations. Overlaid: $\mathcal{B}_i^{\varepsilon_{\min}}$ (black dotted) and $\mathcal{B}_i^R$ (orange dotted), $i = 1, ..., 3$, optimized by solving Probs. 6-$\varepsilon_{\min}$ and 6-R (plans in Fig. 4).

Finally, we can ignore the existential quantifiers in this case: (3d) does not involve θ (as ξ_{xu} does not multiply θ), (3c) implies that $\lambda_{\neg k, i}^j = 0$ for any coordinates i where $g_{\neg k, i}(\xi_j^{\text{loc}}, \theta)$ varies (hence one

4

value, $\lambda = 0$, suffices), and (3a) does not involve $\mathcal{L}$. Thus, a single set of multipliers suffices, and we find the largest box-shaped $\hat{\mathcal{F}}_\theta$ via Prob. 4, a mixed integer second order cone program (MISOCP).

Problem 4 (Box robustification)**.**

$$\underset{s,\theta,\mathcal{L}}{\text{maximize}} \quad \left(\prod_i s_i \right)^{1/d}$$

$$\text{subject to} \quad \{\text{KKT}_{\text{rob}}^{\text{box}}(\xi_j^{\text{loc}})\}_{j=1}^{N_{\text{dem}}}$$

Solving Prob. 4 returns the largest box contained within $\mathcal{F}_\theta$: $\hat{\mathcal{F}}_\theta = \{\theta' \mid \bigwedge_{i=1}^d |\theta_i' - \theta_i| \leq s_i\} \subseteq \mathcal{F}_\theta$. As $\mathcal{F}_\theta$ is a union of boxes, we can extract $\mathcal{F}_\theta$ in its entirety by solving Prob. 4, removing the extracted $\hat{\mathcal{F}}_\theta$ from its feasible set (done with binary variables), and re-solving Prob. 4 with the modified feasible set until it becomes infeasible

(Alg. 1, Fig. 3). Concretely, $\mathcal{F}_\theta = \bigcup_{i=1}^{N_{\text{infeas}}} \hat{\mathcal{F}}_\theta^i$, where $\hat{\mathcal{F}}_\theta^i$ is the box returned at the ith iteration and N_{infeas} is the iteration when infeasibility is reached. We can also prove some theoretical guarantees on Alg. 1 (see App. E for proofs).

Theorem 1. *If Alg. 1 terminates for any parameterization, its output is guaranteed to cover $\mathcal{F}_\theta$.*

Theorem 2. *Alg. 1 is guaranteed to terminate in finite time for union-of-boxes parameterizations.*

Algorithm 1: Iterative $\mathcal{F}_\theta$ extraction

$i = 0$; **while** *Prob. 4 feasible* **do**
$\quad \mid \quad i \leftarrow i+1$; $\hat{\mathcal{F}}_\theta^i \leftarrow$ Prob. 4($\{\hat{\mathcal{F}}_\theta^j\}_{j=1}^{i-1}$);
$\quad \mid \quad$ remove $\hat{\mathcal{F}}_\theta^i$ from Prob. 4's feasible set;
return $\bigcup_i \hat{\mathcal{F}}_\theta^i$

In closing, we refer to App. C.1, where we modify Alg. 1 to more efficiently extract $\mathcal{F}_\theta$ for other constraint parameterizations by covering $\mathcal{F}_\theta$ with zonotopes instead of boxes.

4.2 Obtaining the constraint belief $b(\theta)$

To perform a Bayesian update of $p(\theta)$, conditioning on the extracted $\mathcal{F}_\theta$, we assume that a demonstration is equally likely to have been generated in response to any θ for which it is locally-optimal:

$$p(\{\xi_j^{\text{loc}}\}_{j=1}^{N_{\text{dem}}} \mid \theta) \propto \begin{cases} 1 & \text{if } \{\text{KKT}(\xi_j^{\text{loc}}, \theta)\}_{j=1}^{N_{\text{dem}}} \text{ satisfied} \\ 0 & \text{else} \end{cases} \tag{8}$$

Then, a Bayesian update incorporating the demonstrations amounts to removing all probability mass from KKT-inconsistent θ and renormalizing the probabilities for the KKT-consistent θ:

$$b_{\text{dem}}(\theta) \doteq p(\theta \mid \{\xi_j^{\text{loc}}\}_{j=1}^{N_{\text{dem}}}) = \frac{p(\{\xi_j^{\text{loc}}\}_{j=1}^{N_{\text{dem}}} \mid \theta)p(\theta)}{\int_\Theta p(\{\xi_j^{\text{loc}}\}_{j=1}^{N_{\text{dem}}} \mid \theta)p(\theta)d\theta} = \begin{cases} \frac{p(\theta)}{\int_{\mathcal{F}_\theta} p(\theta)d\theta} & \text{if } \theta \in \mathcal{F}_\theta \\ 0 & \text{else} \end{cases} \tag{9}$$

Finally, we note that this approach is also compatible with uninformative priors (i.e. if no demonstrations are provided) by using the initial prior as the belief: $b(\theta) = p(\theta)$.

5 Policies for adaptive constraint satisfaction

We describe how to use the belief over infinite constraints $b_{\text{dem}}(\theta)$ to plan open-loop trajectories with exact safety probability guarantees (Sec. 5.1), how to plan with more complex constraints with samples from $b_{\text{dem}}(\theta)$ (Sec. 5.2), how $b_{\text{dem}}(\theta)$ can be updated to use constraint data sensed in execution (Sec. 5.3), and how the open-loop plans can be used in a closed-loop policy (Sec. 5.4).

5.1 Planning open-loop trajectories with an *infinite set* of possible constraints

Problem 5. *Chance-constrained plan*

$$\underset{\xi_{xu}}{\min} \quad c_\sigma(\xi_{xu}) \tag{10a}$$

$$\text{s.t.} \quad \bar{\phi}(\xi_{xu}) \in \bar{\mathcal{S}} \subseteq \bar{\mathcal{C}} \tag{10b}$$

$$\phi_\sigma(\xi_{xu}) \in \mathcal{S}_\sigma \subseteq \mathcal{C}_\sigma \tag{10c}$$

$$\Pr(\xi_{xu} \text{ safe}) \geq 1 - \varepsilon \tag{10d}$$

We wish to solve Prob. 5 for convex (10a), which seeks to complete a task while ensuring the plan is safe with probability at least $1 - \varepsilon$ under the belief $b_{\text{dem}}(\theta)$, that is, $\Pr(\xi_{xu} \text{ safe}) = \int_{\Theta_s} b_{\text{dem}}(\theta)d\theta$, where $\Theta_s = \{\theta \mid \phi(\xi_{xu}) \in \mathcal{S}(\theta)\} \subseteq \mathcal{F}_\theta$ is the set of constraints that ξ_{xu} satisfies. Here, the safety threshold $\varepsilon \in [0, 1]$ may be predetermined, or if we wish to plan the safest possible trajectory, we can find

the smallest ε for which Prob. 5 is feasible; denote this variant as Prob. 5-$\varepsilon_{\min}$. Intuitively, Prob. 5 seeks to solve a chance-constrained variant of Prob. 1 for a novel task σ, where the uncertain constraints must be satisfied with a sufficiently high probability. Prob. 5 is challenging due to (10d), as evaluating this probability requires integrating high-dimensional parameters θ over a possibly arbitrarily-shaped Θ_s; hence, (10d) is intractable to enforce exactly for arbitrary distributions and Θ_s. We show that by sacrificing global optimality, it is tractable to enforce (10d) exactly for simple priors $p(\theta)$ by assuming a simple shape for Θ_s.

Our solution, Prob. 6, optimizes over subsets Θ_s that can be represented as a union of boxes $\Theta_s = \bigcup_{i=1}^{N_{\text{boxes}}} \mathcal{B}_i$, $\mathcal{B}_i \subseteq \mathcal{F}_\theta$, for all i (c.f. Sec. 4.1). Each box is parameterized with a center $b_i^{\text{cen}} \in \mathbb{R}^d$ and scalings $b_i^{\text{scale}} \in \mathbb{R}_+^d$: $\mathcal{B}_i = \{b_i^{\text{cen}} + b_i^{\text{scale}} \odot u \mid u \in [-1,1]^d\}$. (11c)-(11f) implement this box-limited chance constraint (see detailed explanations for each constraint in App. A). We restrict focus to priors $p(\theta)$ that can be integrated over boxes in closed form, and

Problem 6. *Riemann-sum chance-constrained plan*

$$\min_{\xi_{xu},\mathcal{B}_i,t_i} c_\sigma(\xi_{xu}) \tag{11a}$$

$$\text{s.t.} \quad \bar{\phi}(\xi_{xu}) \in \bar{\mathcal{S}} \subseteq \bar{\mathcal{C}}, \; \phi_\sigma(\xi_{xu}) \in \mathcal{S}_\sigma \subseteq \mathcal{C}_\sigma \tag{11b}$$

$$\xi_{xu} \in \mathcal{S}(\theta), \; \forall \theta \in \mathcal{B}_1, \ldots, \mathcal{B}_{N_{\text{box}}} \tag{11c}$$

$$\mathcal{B}_i \cap \mathcal{B}_j = \emptyset, \; i \neq j, \quad \mathcal{B}_i \subseteq \mathcal{F}_\theta, \forall i \tag{11d}$$

$$0 \leq t_i \leq (\textstyle\prod_i b_i^{\text{scale}})^{1/d}, \; i = 1, ..., N_{\text{box}} \tag{11e}$$

$$\textstyle\sum_i t_i^d \geq (1-\varepsilon)\text{Vol}(\mathcal{F}_\theta) \tag{11f}$$

for which a monotonic transformation of the resulting integral is concave in b_i^{cen} and b_i^{scale}. While the concavity assumption is satisfied by the broad class of log-concave distributions [33], the closed-form integral is more restrictive. In this paper, we focus only on a uniform $p(\theta)$ (see App. D.3 for extensions to other distributions); note that this does *not* imply a uniform probability of safety over the constraint space $\mathcal{C}$. Intuitively, Prob. 6 performs a box-limited Riemann sum integration over the constraint belief. Each box $\mathcal{B}_i$ represents a subset of $\mathcal{F}_\theta$ over which the probability is integrated (c.f. Fig. 3). For piecewise affine (PWA) dynamics, Prob. 6 can be written as an MISOCP, except for (11f) which renders Prob. 6 an MIBLP: solvable with [34], but possibly slow. We can replace (11f) with a linear surrogate $\sum_i t_i \geq (1-\varepsilon)\text{Vol}(\mathcal{F}_\theta)$, but this can still be slow if N_{box} is large. We discuss efficient reformulations of Prob. 6 in App. D. Overall, we have this result (proof in App. E):

Theorem 3. *A solution to Prob. 6 is a guaranteed feasible, possibly suboptimal solution to Prob. 5.*
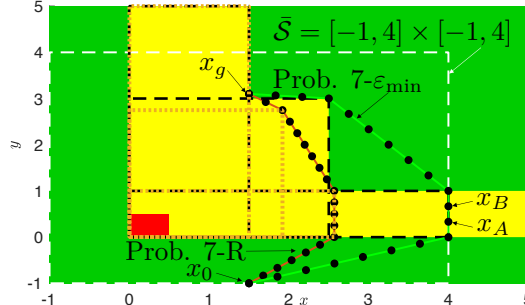

Fig. 4: Generated plans for $\mathcal{B}_i$ in Fig. 3. $\text{proj}_{\mathcal{X}}(\mathcal{B}_i)$ are overlaid (with matched color).

Instead of the chance-constrained formulation, we can directly trade off the cost and the safety probability in the objective; i.e. in Probs. 5; 6, change (10a); (11a) to the ratio $c_\sigma(\xi_{xu})/\Pr(\xi_{xu} \text{ safe})$; $c_\sigma(\xi_{xu})/\sum_i t_i^d$ and remove (10d); (11f). Denote these variants Prob. 5-R; Prob. 6-R. Note that after linearizing $\sum_i t_i^d$, the objective is quasi-convex, so we can rewrite Prob. 6-R as a feasibility problem with a new optimization constraint $c_\sigma(\xi_{xu}) - \alpha\Pr(\xi_{xu} \text{ safe}) \leq 0$, and do a line search on α, solving Prob. 6-R as a sequence of MISOCPs.

5.2 Planning open-loop trajectories with a *finite set* of sampled possible constraints

For complex constraints arising from nonlinear dynamics or constraint parameterizations, Prob. 6 is hard to solve as it involves both integer variables and nonlinearities. While sometimes we can plan for a PWA model that roughly captures the nonlinear dynamics (i.e. modeling a quadrotor as a double integrator), we generally use sampled approximations of Prob. 6 when it cannot be written as a mixed integer convex program (MICP), in particular, Minimum Constraint Removal (MCR) [35] and the Blindfolded Traveler's Problem (BTP) [36], briefly described here (c.f. App. D for details):

I. MCR takes a *finite* set of constraints and incrementally constructs a roadmap to connect a start and goal state while violating the minimum number of constraints. MCR can be used to approximate Prob. 6-$\varepsilon_{\min}$ by sampling a finite set of constraints $\{\theta_i \sim b(\theta)\}_{i=1}^{N_{\text{sam}}}$ as input to MCR.

II. BTP is a roadmap-based planner for additive cost functions $c_\sigma(\xi_{xu})$ which takes as input a state space graph (V, E), where executing edge $e \in E$ costs $c(e)$ with probability $p(e)$ of being safe. To use BTP, we sample constraints $\{\theta_i \sim b(\theta)\}_{i=1}^{N_{sam}}$, and use them to approximate $p(e)$; we plan on the graph by running A* with modified edge costs $\sum_e c_\sigma(e) - \beta p(e \text{ safe})$, for some weight β.

We note that while the sampled approximations can be more flexible than Prob. 6, they are not guaranteed to return a feasible solution to Prob. 5, as it depends on the constraints that are sampled.

5.3 Updates to $b(\theta)$ in online execution

Our framework can also incorporate uncertain information about the true constraint sensed in execution by computing a belief update. Suppose that we are given $\mathfrak{C}_s \doteq \{\mathfrak{C}_s^i\}_{i=1}^{N_s}$ and $\mathfrak{C}_{\neg s} \doteq \{\mathfrak{C}_{\neg s}^i\}_{i=1}^{N_{\neg s}}$ as a set of *sets of possibly safe/unsafe states*, respectively, where each $\mathfrak{C}_s^i$ / $\mathfrak{C}_{\neg s}^i$ denotes a finite set where at least one state is safe/unsafe. Let the set of all constraints consistent with $\mathfrak{C}_s, \mathfrak{C}_{\neg s}$ be $\mathcal{F}_\theta^{s, \neg s} \doteq \{\theta \in \mathcal{F}_\theta \mid \bigwedge_{i=1}^{N_s}(\exists \kappa \in \mathfrak{C}_s^i, g(\kappa, \theta) \leq 0) \wedge \bigwedge_{i=1}^{N_{\neg s}}(\exists \kappa \in \mathfrak{C}_{\neg s}^i, g(\kappa, \theta) > 0)\}$. We compute $\mathcal{F}_\theta^{s, \neg s}$ iteratively with a variant of Alg. 1 (see App. D.4 for details). Finally, we perform the update:

$$b_{ex}(\theta) \doteq p(\theta \mid \{\xi_j^{loc}\}_{j=1}^{N_{dem}}, \mathcal{C}_s, \mathcal{C}_{\neg s}) = \begin{cases} \frac{p(\theta)}{\int_{\mathcal{F}_\theta^{s, \neg s}} p(\theta) d\theta} & \text{if } \theta \in \mathcal{F}_\theta^{s, \neg s} \\ 0 & \text{else} \end{cases} \quad (12)$$

This setup can handle data from many different constraint sensing modalities, like direct, exact sensing (from a bump sensor), long-range measurements (from bounded-range LiDAR), or uncertain contact measurements [36] where collision cannot be exactly localized on the robot volume (see App. D.4 for more details).
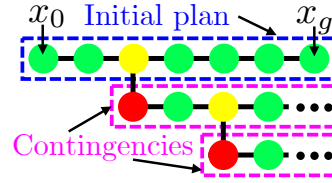

Fig. 5: Policy tree: initial plan and contingencies, rooted at possibly unsafe states. Green / red / yellow states in $\mathcal{G}_s$ / $\mathcal{G}_{\neg s}$ / $\text{proj}_\mathcal{C}(\mathcal{F}_\theta)$.

5.4 Closed-loop policies for adaptive constraint satisfaction

Finally, we compute closed-loop policies for Probs. MCV and MEC. Our strategy is simple: for Probs. MCV and Prob. MEC, solve Prob. 6-ε_{min} and Prob. 6-R, respectively, in a receding horizon fashion, i.e at each time-step, we update $b(\theta)$ with the new constraint information, re-solve the optimization, and switch to the new solution if the previous plan is suboptimal/unsafe. This policy takes $p(\theta)$ and a sequence of observations $\mathcal{O} = (\mathfrak{C}_s, \mathfrak{C}_{\neg s})$ to estimate a belief, and uses the belief and current state to output u. These strategies are motivated by results in sequential decision making that provide approximation guarantees for greedy policies proposing solutions that minimize the ratio of cost to safety probability at each iteration [37]; Probs. 6-ε_{min}, 6-R both do this.

Note that as our policy executes a plan until sensing implies it is suboptimal or unsafe, upon which it switches, it can be represented as a tree, where contingency plans are rooted only at states on the current plan where switches can occur. See Fig. 5 for the simple case where the policy only switches upon learning the current plan is unsafe; here, the branching is sparse, occurring only at possibly unsafe points (yellow) on the initial and replanned trajectories. In these cases, we can exploit the sparsity to avoid solving Prob. 6 at runtime by precomputing the contingency plans, facilitating real-time policy execution. For tractability, the precomputation assumes no unmodeled obstacles appear at runtime (this eliminates the sparse branching, as it makes each state on the plan possibly unsafe), discretizes the set of possible continuous sensing measurements (if not, we could need to compute contingencies for an infinite set of possible measurements), and terminates branching at a finite tree depth (as planning may be infeasible for a worst-case constraint). If the assumptions do not hold (as in some of our results), we can always compute new plans online, though it can be slow.

As an example, consider computing contingencies for the green plan in Fig. 4. Only x_A and x_B lie in the possibly unsafe (yellow) region, so if no unmodeled obstacles appear at runtime, we can only be forced to replan in two cases: 1) x_A is unsafe, 2) x_A is safe and x_B is unsafe. In case 1, we

update the belief, keeping only constraints marking x_A as unsafe, and plan a contingency satisfying as many constraints as possible from the new belief. This repeats recursively, up to a finite recursion depth, for any possibly unsafe states on the contingency. In case 2, updating the belief to mark x_A as safe and x_B as unsafe renders the belief empty, since this is impossible given the initial belief and box parameterization. We can thus avoid computing further contingencies on this policy tree branch.

Problem Constraints/prior	Prob. MCV	Prob. MEC
MICP-representable	Prob. 6-$\varepsilon_{\min}$	Prob. 6-R
Not MICP-rep.	MCR	BTP

Table 1: Which open-loop planner to use?

To recap, we would ideally solve Prob. 5 to get open-loop plans for our policy, but it is intractable. If all constraints (dynamics, uncertain constraints, etc.) and the integrals of $p(\theta)$ are MICP-representable, we can approximate Prob. 5 with Prob. 6, which enjoys theoretical guarantees by using the *infinite* belief. If not, we use MCR/BTP, which use *finite samples* from the belief. This is summarized in Table 1.

6 Experiments

We show our method scales to safely and efficiently plan for high-dimensional (12D) systems with combined state/control constraint uncertainty, constraint sensing uncertainty, and high-dimensional (30D) constraints. See App. F for more details and experiments (7-DOF arm planning with suboptimal demonstrations, and nonlinear constraint planning using zonotope-based $\mathcal{F}_\theta$ extraction), App. F.6 for computation time discussion, and https://youtu.be/aWZ_U-gWQJI for visualizations.

Mixed quadrotor uncertainty: We plan for a quadrotor (dynamics in App. F) carrying a payload of uncertain weight around uncertain obstacles. We are given one demonstration (Fig. 6.B, pink) to learn a 7D constraint $\theta \in \mathbb{R}^7$ (6 for obstacle, 1 for control). After extracting $\mathcal{F}_\theta$ with Alg. 1, (Fig. 6.B), we remain uncertain about the obstacle's y-extents. We model the uncertain weight as an unknown control constraint $\|u\|_2^2 \leq \bar{U}(\theta)$; from the demonstration, we learn that $\|u\|_2^2 \leq 97.85$ is guaranteed safe (Fig. 6.A), and KKT also tells us the constraint is inactive, so $\|u\|_2^2 \in (97.85, 100] = \text{proj}_{\mathcal{U}}(\mathcal{F}_\theta)$ is possibly unsafe. The quadrotor has bump and torque sensors to directly detect state and control constraint violations, respectively. We now solve Prob. MCV starting from a lower initial state (Fig. 6.C), which we do by solving Prob. 6-$\varepsilon_{\min}$ for an initial plan and contingencies, directly optimizing over the infinite constraint belief. We use a double-integrator approximation of the quadrotor dynamics and restrict each open-loop trajectory to 30 timesteps; thus, it is not possible to satisfy all constraints in $\mathcal{F}_\theta$ while reaching the goal in the time limit. Solving Prob. 6-$\varepsilon_{\min}$ returns Plan 1 (Fig. 6.C), which violates some possible control constraints in order to lift the quadrotor over all possible obstacles. Our policy also generates contingencies (Fig. 6.D-F) in case Plan 1 violates the true control constraint, and we must plan to avoid the possible obstacles. To emphasize the benefit of *optimizing* over the infinite set of possible constraints, we compare to two baselines: a variant of the scenario approach [26], where we iteratively *sample* and enforce $\{\theta_i \sim b(\theta)\}_{i=1}^{N_{\text{sam}}}$
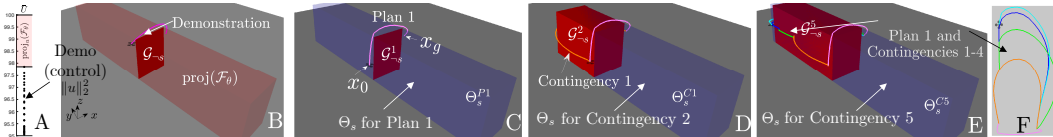

Fig. 6: Mixed quadrotor uncertainty example. **A-B**. Initial control and state constraint uncertainty. **C**. Initial plan for a new task. **D-F**. Contingencies are pre-computed, and the system switches if the initial plan is unsafe.

7-DOF arm with contact sensing uncertainty: We plan for a 7-DOF arm (dynamics in App. F) near a storage rack. We are given two demonstrations (Fig. 7.A), and after running Alg. 1 to obtain

8

$b_{\text{dem}}(\theta)$, partly reveal the shelf constraint (Fig. 7.B), which has parameters $\theta \in \mathbb{R}^6$. We assume an uncertain contact sensing model [38], where contact is assumed to be at any point on the arm downstream on the kinematic chain where a torque limit is exceeded (c.f. App. D). We solve Prob. MEC for $c_\sigma(\xi_{xu}) = \sum_{t=1}^{T-1} \|x_{t+1} - x_t\|_2$, for the task of moving the arm from below to above the shelf (Fig. 7.B), using 100 (uniform) samples from $b_{\text{dem}}(\theta)$ in BTP. We compare our policy to 1) BTP without demonstrations and the union-of-boxes parameterization, and 2) an optimistic approach that executes the optimal path on the BTP graph after removing unsafe edges, and replanning if a constraint is violated. From the demonstrations, we can determine that a subset of the shelf is guaranteed unsafe (Fig. 7, dark red); beyond that, we are uncertain (Fig. 7.B). Thus, we plan to avoid that region, swinging the arm around and over the uncertain area. However, in doing so, the arm bumps into an unmodeled obstacle: a box on the lower shelf (Fig. 7.C). The contact sensor informs our method that some point on the end effector is in collision; we add 300 sampled points on the end effector to $\mathfrak{C}_{\neg s}$ and samples from the traversed free space to $\mathfrak{C}_s$. Our method then automatically determines that it needs to update the constraint parameterization, as geometrically there is no single box that can explain the demonstrations, $\mathfrak{C}_s$, and $\mathfrak{C}_{\neg s}$. After updating θ to include two boxes (now $\theta \in \mathbb{R}^{12}$), we extract $\mathcal{F}_\theta$ for this updated parameterization, and resample 100 (uniform) samples from the updated $b_{\text{ex}}(\theta)$ as input to BTP (see Fig. 7.D). As our belief now indicates an uncertain region near the lower shelf obstacle, our policy plans to move further out from the shelf to avoid the uncertain region, and reaches the goal. Running Alg. 1 and BTP takes 20 and between 5-20 min. (can be sped up by precomputing arm swept volumes, see [36], App. F.6), respectively. Overall, our
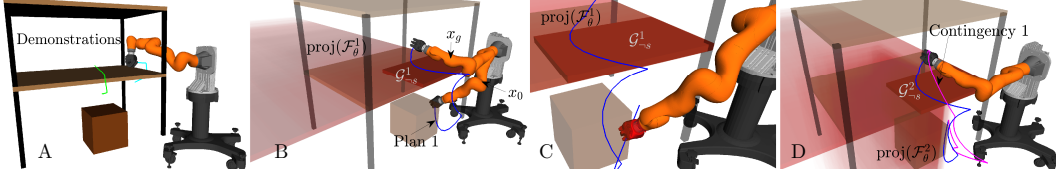

Fig. 7: Arm with contact sensing uncertainty. **A**: Demonstrations. **B**. Initial constraint uncertainty (red) and plan (blue). **C**. The initial plan violates an unmodeled constraint, triggering a belief update. **D**. Replan online.

Quadrotor maze: We plan for a quadrotor in clutter with two-meter radius LiDAR sensing (Fig. 8). We know the brown obstacles *a priori*, and are given five demonstrations that reveal five obstacles ($\theta \in \mathbb{R}^{30}$) (Fig. 8.A), but provide little information about their size. We solve Prob. MEC steering from the bottom to the top of the maze (Fig. 8.D) while minimizing $c(\xi_{xu}) = \sum_{t=1}^{T-1} \|u_t\|_2^2$ by solving Prob. 6-R, optimizing directly over the continuous $b(\theta)$, and computing contingencies. We also modify Prob. 6-R to never collide in execution by avoiding the set of inevitable collision states [39] under the double-integrator model; this is modeled with additional constraints (see App. D). We obtain dynamically-feasible quadrotor trajectories by warmstarting the nonlinear optimization with the double-integrator trajectory. We visualize our policy in Fig. 8. Running Alg. 1 and Prob. 6 takes 1 sec. and 1 min., respectively. Plan 1 (pink) intelligently trades off risk and performance. Note there may exist a direct path to the goal between the brown obstacles; however, the orange demonstration induces an obstacle that likely blocks this path. Also, moving left is riskier than moving right, as the uncertain obstacle on the left may create a dead end. Plan 1 avoids both traps, moving to the right and increasing altitude to avoid all possible obstacles induced by the dark blue demonstration. This enables maintenance of higher speed and thus lower cost, instead of cautiously approaching the uncertain region to determine if it is safe to cut through. Finally, Plan 1 cuts through the possibly-unsafe region induced by the green demonstration, as the obstacle is unlikely to extend down to the brown obstacle. We discretize the possible constraint measurements in this region on a grid, planning contingencies if the passage is partially (Contingencies 1-3) or completely blocked (4). We compare to two approaches, [1], which plans trajectories which are guaranteed-safe under the constraint parameterization, and [27], which plans optimistically over sets of subgoals on the frontier (see App. F.1 for more details). Drawing constraints uniformly from $\mathcal{F}_\theta$, our policy solves
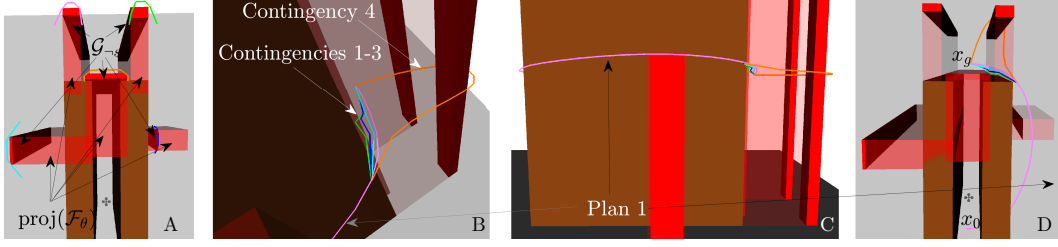
Fig. 8: Quadrotor maze. **A**. Demos., initial constraint uncertainty. **B-D**. Three views of the initial plan (pink) and contingencies for different sensing possibilities, obtained by gridding the possible sensor measurements.

7 Conclusion

We present a method to address uncertainty in constraints learned from demonstrations. Instead of trying to satisfy all possible constraints, we obtain a belief over constraints, then design open-loop planners which use the belief to "be as safe as possible while remaining efficient". We use these planners in a closed-loop policy that uses constraint data gathered online to help complete the task. In future work, we aim to speed up our method with parallel extraction and fast integer programming [40], and extend our method to adaptively plan with beliefs over temporal logic formulas [41].

References

[1] G. Chou, N. Ozay, and D. Berenson. Learning constraints from locally-optimal demonstrations under cost function uncertainty. *IEEE Robotics Autom. Lett.*, 5(2):3682–3690, 2020.

[2] G. Chou, D. Berenson, and N. Ozay. Learning constraints from demonstrations. *Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2018. URL arxiv.org/abs/1812.07084.

[3] M. Menner, P. Worsnop, and M. N. Zeilinger. Constrained inverse optimal control with application to a human manipulation task. *IEEE TCST*, 2019.

[4] P. Englert, N. A. Vien, and M. Toussaint. Inverse kkt: Learning cost functions of manipulation tasks from demonstrations. *IJRR*, 36(13-14):1474–1488, 2017.

[5] G. Chou, N. Ozay, and D. Berenson. Learning parametric constraints in high dimensions from demonstrations. In *CoRL 2019, Osaka, Japan*, pages 1211–1230, 2019.

[6] C. Li and D. Berenson. Learning object orientation constraints and guiding constraints for narrow passages from one demonstration. In *ISER*, 2016.

[7] C. Pérez-D'Arpino and J. A. Shah. C-LEARN: learning geometric constraints from demonstrations for multi-step manipulation in shared autonomy. In *ICRA*, 2017.

[8] S. Calinon and A. Billard. A probabilistic programming by demonstration framework handling constraints in joint space and task space. In *IROS*, 2008. doi:10.1109/IROS.2008.4650593.

[9] A. L. Pais, K. Umezawa, Y. Nakamura, and A. Billard. Learning robot skills through motion segmentation and constraints extraction. HRI, 2013.

[10] L. Armesto, J. Bosga, V. Ivan, and S. Vijayakumar. Efficient learning of constraints and generic null space policies. In *ICRA*, 2017.

[11] D. Park, M. Noseworthy, R. Paul, S. Roy, and N. Roy. Inferring task goals and constraints using bayesian nonparametric inverse reinforcement learning. In *CoRL*, pages 1005–1014, 2019.

[12] M. Bujarbaruah, C. Vallon, and F. Borrelli. Learning to satisfy unknown constraints in iterative MPC. *CoRR*, abs/2006.05054, 2020. URL https://arxiv.org/abs/2006.05054.

[13] A. Shah, S. Li, and J. Shah. Planning with uncertain specifications (puns). *IEEE Robotics Autom. Lett.*, 5(2):3414–3421, 2020.

[14] N. D. Ratliff, J. A. Bagnell, and M. Zinkevich. Maximum margin planning. In *ICML*, 2006.

[15] P. Abbeel and A. Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *ICML*, 2004. doi:10.1145/1015330.1015430.

[16] D. Ramachandran and E. Amir. Bayesian inverse reinforcement learning. In *IJCAI*, 2007.

[17] S. Ross, G. J. Gordon, and D. Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *AISTATS*, pages 627–635, 2011.

[18] Y. Cui, D. Isele, S. Niekum, and K. Fujimura. Uncertainty-aware data aggregation for deep imitation learning. In *ICRA 2019*, pages 761–767, 2019.

[19] D. S. Brown, R. Coleman, R. Srinivasan, and S. Niekum. Safe imitation learning via fast bayesian reward inference from preferences. *ICML*, 2020.

[20] K. Menda, K. R. Driggs-Campbell, and M. J. Kochenderfer. Ensembledagger: A bayesian approach to safe imitation learning. In *IROS*, pages 5041–5048, 2019.

[21] J. Zhang and K. Cho. Query-efficient imitation learning for end-to-end autonomous driving. *CoRR*, abs/1605.06450, 2016. URL http://arxiv.org/abs/1605.06450.

[22] S. Thakur, H. van Hoof, J. C. G. Higuera, D. Precup, and D. Meger. Uncertainty aware learning from demonstrations in multiple contexts using bayesian neural networks. In *ICRA*, 2019.

[23] S. Choi, K. Lee, S. Lim, and S. Oh. Uncertainty-aware learning from demonstration using mixture density networks with sampling-free variance modeling. In *ICRA 2018*.

[24] B. Axelrod, L. P. Kaelbling, and T. Lozano-Pérez. Provably safe robot navigation with obstacle uncertainty. In *R:SS XIII*, 2017.

[25] M. P. Vitus, Z. Zhou, and C. J. Tomlin. Stochastic control with uncertain parameters via chance constrained control. *IEEE Transactions on Automatic Control*, 61(10):2892–2905, 2016.

[26] S. Grammatico, X. Zhang, K. Margellos, P. J. Goulart, and J. Lygeros. A scenario approach for non-convex control design. *IEEE Trans. Autom. Control.*, 61(2):334–345, 2016.

[27] L. Janson, T. Hu, and M. Pavone. Safe motion planning in unknown environments: Optimality benchmarks and tractable policies. In *R:SS XIV*, 2018.

[28] C. Richter, W. Vega-Brown, and N. Roy. Bayesian learning for safe high-speed navigation in unknown environments. In *ISRR*, pages 325–341, 2015.

[29] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[30] T. Tao. *Analysis II*. Springer, 3rd edition, 2016.

[31] A. Ben-Tal, L. E. Ghaoui, and A. Nemirovski. *Robust Optimization*, volume 28 of *Princeton Series in Applied Mathematics*. Princeton University Press, 2009.

[32] F. Alizadeh and D. Goldfarb. Second-order cone programming. *Math. Program.*, 95(1), 2003.

[33] M. Bagnoli and T. Bergstrom. Log-concave probability and its applications. *Economic Theory*, 26(2):445–469, Aug 2005.

[34] L. Gurobi Optimization. Gurobi optimizer reference manual, 2020.

[35] K. K. Hauser. The minimum constraint removal problem with three robotics applications. *IJRR*, 33(1):5–17, 2014.

[36] B. Saund, S. Choudhury, S. Srinivasa, and D. Berenson. The blindfolded robot: A bayesian approach to planning with contact feedback. In *ISRR*, 2019.

[37] A. Dor, E. Greenshtein, and E. Korach. Optimal and myopic search in a binary random vector. *Journal of Applied Probability*, 35(2), 1998.

[38] B. Saund and D. Berenson. Motion planning for manipulators in unknown environments with contact sensing uncertainty. In *ISER*, pages 461–474, 2018.

[39] T. Fraichard and H. Asama. Inevitable collision states - a step towards safer robots? *Adv. Robotics*, 18(10):1001–1024, 2004.

[40] D. Bertsimas and B. Stellato. Online mixed-integer optimization in milliseconds. *CoRR*, abs/1907.02206, 2019. URL http://arxiv.org/abs/1907.02206.

[41] G. Chou, N. Ozay, and D. Berenson. Explaining Multi-stage Tasks by Learning Temporal Logic Formulas from Suboptimal Demonstrations. In *Proceedings of Robotics: Science and Systems*, Corvalis, Oregon, USA, July 2020. doi:10.15607/RSS.2020.XVI.097.

[42] M. Herceg, M. Kvasnica, C. Jones, and M. Morari. Multi-Parametric Toolbox 3.0. In *ECC*, 2013.

[43] J. Bochnak, M. Coste, and M.-F. Roy. *Real Algebraic Geometry*. Springer, 1998.

[44] G. E. Collins. Hauptvortrag: Quantifier elimination for real closed fields by cylindrical algebraic decomposition. In *Automata Theory and Formal Languages*, pages 134–183, 1975.

[45] V. Magron, D. Henrion, and J. Lasserre. Semidefinite approximations of projections and polynomial images of semialgebraic sets. *SIAM Journal on Optimization*, 25(4):2143–2164, 2015.

[46] M. Beck and S. Robins. *Computing the Continuous Discretely: Integer-Point Enumeration in Polyhedra*, pages 167–182. Springer New York, New York, NY, 2015.

[47] A. Ben-Tal, A. P. Goryashko, E. Guslitzer, and A. Nemirovski. Adjustable robust solutions of uncertain linear programs. *Math. Program.*, 99(2):351–376, 2004.

[48] F. Sabatino. Quadrotor control: modeling, nonlinear control design, and simulation, 2015.

Appendix

In these appendices, we will first summarize and provide more details on the optimization problems used in our method (Appendix A, discuss various results on the representability of sets of cost function and constraint parameters which are consistent with demonstrations (Appendix B), provide expanded details on our methods for extracting the set of consistent cost function and constraint parameters (Appendix C), provide expanded details on our methods for planning policies for adaptively satisfying uncertain constraint parameters (Appendix D), provide proofs for the theoretical results in the main body (Appendix E), and provide additional details on our experimental results (Appendix F).

A Optimization problem glossary

In this appendix, we provide a detailed summary of the optimization problems utilized in our approach.

Problem 1: This is the optimization problem that we assume the demonstrator is solving to local-optimality. This problem involves a potentially task-dependent cost function $c_\sigma(\xi_{xu})$, where a task in this paper is simply steering the system state from a start state x_0 to a goal state x_g while satisfying a set of constraints. This problem also involves a known shared constraint $\bar\phi(\xi_{xu}) \in \bar{\mathcal{S}}$ (which embeds known constraints which that shared across all tasks, such as the system dynamics) as well as a known task-dependent constraint $\phi_\sigma(\xi_{xu}) \in \mathcal{S}_\sigma$ (which embeds known constraints that are task-dependent, such as the start and goal state constraints). Finally, there is the unknown shared constraint $\phi(\xi_{xu}) \in \mathcal{S}(\theta)$ (unknown to the learner, but known to the demonstrator) which is parameterized by unknown parameters θ.

$$
\begin{aligned}
&\underset{\xi_{xu}}{\text{minimize}} && c_\sigma(\xi_{xu}) \\
&\text{subject to} && \phi(\xi_{xu}) \in \mathcal{S}(\theta) \subseteq \mathcal{C} && \Leftrightarrow && \mathbf{g}_{\neg k}(\xi_{xu}, \theta) \leq \mathbf{0} \\
& && \bar\phi(\xi_{xu}) \in \bar{\mathcal{S}} \subseteq \bar{\mathcal{C}}, \quad \phi_\sigma(\xi_{xu}) \in \mathcal{S}_\sigma \subseteq \mathcal{C}_\sigma && \Leftrightarrow && \mathbf{h}_k(\xi_{xu}) = \mathbf{0}, \quad \mathbf{g}_k(\xi_{xu}) \leq \mathbf{0}
\end{aligned}
$$

Problem 2: This is the inverse optimization problem that the learner solves to learn *one possible assignment* of the unknown constraints that are satisfied by the demonstrations. Specifically, the problem searches over the unknown constraint parameters and the Lagrange multipliers which together make the KKT conditions of each demonstration satisfied.

$$
\begin{aligned}
&\text{find} && \theta, \mathcal{L} \doteq \{\boldsymbol{\lambda}_k^j, \boldsymbol{\lambda}_{\neg k}^j, \boldsymbol{\nu}_k^j\}_{j=1}^{N_{\text{dem}}} \\
&\text{subject to} && \{\text{KKT}(\xi_j^{\text{loc}})\}_{j=1}^{N_{\text{dem}}}
\end{aligned}
$$

Problem 3: This problem returns the set of all consistent constraint parameters $\mathcal{F}_\theta$. Intuitively, this problem finds the largest set $\hat{\mathcal{F}}_\theta$, in the sense of set containment, of constraint parameters θ, such that the KKT conditions can be made to hold for those parameters. The problem is intractable in its most general form, motivating simpler variants Problem 4 and Problem 7.

$$
\begin{aligned}
&\underset{\hat{\mathcal{F}}_\theta}{\sup} && \text{Vol}(\hat{\mathcal{F}}_\theta) \\
&\text{s.t.} && \forall \theta \in \hat{\mathcal{F}}_\theta, \;\; \exists\{\boldsymbol{\lambda}_k^j, \boldsymbol{\lambda}_{\neg k}^j, \boldsymbol{\nu}_k^j \mid \text{KKT}(\xi_j^{\text{loc}})\}_{j=1}^{N_{\text{dem}}}
\end{aligned}
$$

Problem 4: This problem returns the largest axis-aligned hyper-rectangle contained within $\mathcal{F}_\theta$; this problem is a restricted, tractable version of Problem 3.

$$
\begin{aligned}
&\underset{s, \theta, \mathcal{L}}{\text{maximize}} && \left(\textstyle\prod_i s_i\right)^{1/d} \\
&\text{subject to} && \{\text{KKT}_{\text{rob}}^{\text{box}}(\xi_j^{\text{loc}})\}_{j=1}^{N_{\text{dem}}}
\end{aligned}
$$

Problem 5: This problem solves a chance-constrained trajectory optimization problem, minimizing a possibly task-dependent objective $c_\sigma(\xi_{xu})$ while ensuring that the resulting trajectory satisfies the uncertain constraint with prescribed probability $1 - \varepsilon$. We further consider two specific variants, **Problem 5-$\varepsilon_{\mathbf{min}}$**, which seeks to solve Problem 5 to be as safe as possible, i.e with the smallest ε for

13

which there exists a feasible solution, and **Problem 5-R**, which directly trades off performance and safety with a modified objective $c_\sigma(\xi_{xu})/\Pr(\xi_{xu} \text{ safe})$. Problem 5 and its variants are in their most general form intractable, motivating the simpler variant Problem 6.

Problem 5:

$$\min_{\xi_{xu}} \quad c_\sigma(\xi_{xu}) \tag{13a}$$

$$\text{s.t.} \quad \bar{\phi}(\xi_{xu}) \in \bar{\mathcal{S}} \subseteq \bar{\mathcal{C}} \tag{13b}$$

$$\phi_\sigma(\xi_{xu}) \in \mathcal{S}_\sigma \subseteq \mathcal{C}_\sigma \tag{13c}$$

$$\Pr(\xi_{xu} \text{ safe}) \geq 1 - \varepsilon \tag{13d}$$

Problem 5-$\varepsilon_{\min}$:

$$\min_{\xi_{xu}} \min_{\varepsilon} \quad c_\sigma(\xi_{xu})$$

$$\text{s.t.} \quad \bar{\phi}(\xi_{xu}) \in \bar{\mathcal{S}} \subseteq \bar{\mathcal{C}}$$

$$\phi_\sigma(\xi_{xu}) \in \mathcal{S}_\sigma \subseteq \mathcal{C}_\sigma$$

$$\Pr(\xi_{xu} \text{ safe}) \geq 1 - \varepsilon$$

Problem 5-R:

$$\min_{\xi_{xu}} \quad c_\sigma(\xi_{xu})/\Pr(\xi_{xu} \text{ safe})$$

$$\text{s.t.} \quad \bar{\phi}(\xi_{xu}) \in \bar{\mathcal{S}} \subseteq \bar{\mathcal{C}}$$

$$\phi_\sigma(\xi_{xu}) \in \mathcal{S}_\sigma \subseteq \mathcal{C}_\sigma$$

Problem 6: This is a simplified variant of Problem 5, which makes the probability constraint tractable by restricting the integration of probability mass over a fixed number of axis-aligned boxes, where the location and extents of the boxes are also optimized over. We consider two specific variants, **Problem 6-$\varepsilon_{\min}$** and **Problem 6-R**, which are as described for Problem 5.

$$\min_{\xi_{xu}, \mathcal{B}_i, t_i} \quad c_\sigma(\xi_{xu}) \tag{16a}$$

$$\text{s.t.} \quad \bar{\phi}(\xi_{xu}) \in \bar{\mathcal{S}} \subseteq \bar{\mathcal{C}}, \ \phi_\sigma(\xi_{xu}) \in \mathcal{S}_\sigma \subseteq \mathcal{C}_\sigma \tag{16b}$$

$$\xi_{xu} \in \mathcal{S}(\theta), \ \forall \theta \in \mathcal{B}_1, \ldots, \mathcal{B}_{N_{\text{box}}} \tag{16c}$$

$$\mathcal{B}_i \cap \mathcal{B}_j = \emptyset, \ i \neq j, \quad \mathcal{B}_i \subseteq \mathcal{F}_\theta, \forall i \tag{16d}$$

$$0 \leq t_i \leq \left(\prod_i b_i^{\text{scale}} \right)^{1/d}, \ i = 1, \ldots, N_{\text{box}} \tag{16e}$$

$$\sum_i t_i^d \geq (1 - \varepsilon) \text{Vol}(\mathcal{F}_\theta) \tag{16f}$$

We provide an overview of the constraints of Problem 6. First, note that (16a)-(16b) exactly correspond to (13a)-(13c). The remaining constraints in Problem 6 implement the box-limited integration. Specifically, (16c) enforces that the planned trajectory ξ_{xu} is safe with respect to all θ belonging to $\mathcal{B}_1, \ldots, \mathcal{B}_{N_{\text{box}}}$. Recall that each $\mathcal{B}_i$ is meant to represent a box contained in $\mathcal{F}_\theta$, and that ξ_{xu} is to be safe with respect to all θ belonging to this $\mathcal{B}_i$. Thus, (16d) further enforces that each $\mathcal{B}_i$ is contained in $\mathcal{F}_\theta$, and furthermore, that the $\mathcal{B}_i$ are disjoint; this is to avoid any double-counting of box volumes (and thus probability mass). Next, (16e) introduces the variables t_i, from which the volume of the corresponding box can be recovered: note that the volume of $\mathcal{B}_i$ is $\prod_i b_i^{\text{scale}}$. The last constraint, (16f), does exactly this: t_i^d is exactly the volume of $\mathcal{B}_i$, so $\sum_i t_i^d = \sum_i \text{Vol}(\mathcal{B}_i)$, which we ensure is at least $(1 - \varepsilon)\text{Vol}(\mathcal{F}_\theta)$ to satisfy the probability constraint.

Problem 7: In Appendix C.1, we will discuss a modification of Algorithm 1 which makes it more efficient for constraint parameterizations other than the union-of-boxes parameterization assumed in Sec. 4. This modification hinges upon Problem 7, which returns a zonotope contained within $\mathcal{F}_\theta$ of approximately maximum volume; this problem is a restricted, tractable version of Problem 3 which is more general than Problem 4.

$$\underset{s, \theta, \boldsymbol{\lambda}_k^j, \boldsymbol{\lambda}_{\neg k}^j, \boldsymbol{\nu}_k^j, Q_i}{\text{maximize}} \quad \sum_{i=1}^{N_{\text{gen}}} \|\ell_i\|_1$$

$$\text{subject to} \quad \{\text{KKT}_{\text{rob}}^{\text{zon}}(\xi_j^{\text{loc}})\}_{j=1}^{N_{\text{dem}}}, \quad |\ell_m^\top \ell_n| \leq \delta, \ \forall m \neq n$$

B A geometric analysis of constrained inverse optimal control

We describe how the constraint learning problem can be extended to also learn unknown cost function parameters γ (Appendix B.1), the shape of the resulting feasible sets for unknown cost function parameters for various parameterizations (Appendix B.2), and the shape of the resulting feasible sets for consistent constraint parameters (Appendix B.3), for various parameterizations.

B.1 Modifying Problem 2 to handle unknown cost function parameters

As in [1], we note that the KKT conditions in Problem 2 can be modified to handle unknown cost function parameters, where the cost function can be written as $c_\sigma(\xi_{xu}, \gamma)$ for unknown cost function parameters $\gamma \in \Gamma$, with few changes: the only KKT condition that changes is stationarity (3d), where the term involving the gradient of the cost now also involves the unknown cost function parameters γ:

$$\nabla_{\xi_{xu}} c_\sigma(\xi_j^{\text{loc}}, \gamma) + \boldsymbol{\lambda}_k^{j\top} \nabla_{\xi_{xu}} \mathbf{g}_k(\xi_j^{\text{loc}}) + \boldsymbol{\lambda}_{\neg k}^{j\top} \nabla_{\xi_{xu}} \mathbf{g}_{\neg k}(\xi_j^{\text{loc}}, \theta) + \boldsymbol{\nu}_k^{j\top} \nabla_{\xi_{xu}} \mathbf{h}_k(\xi_j^{\text{loc}}) = \mathbf{0} \quad (17)$$

B.2 Unknown cost function, known constraint

Let us denote the feasible set of Problem 2, modified to handle unknown cost function parameters, again be $\mathcal{F}$, and let us denote the projection of $\mathcal{F}$ onto Γ as $\mathcal{F}_\gamma$:

$$\mathcal{F}_\gamma \doteq \{\gamma \mid \exists (\theta, \mathcal{L}) : (\theta, \gamma, \mathcal{L}) \in \mathcal{F}\} \quad (18)$$

In the following, we will analyze the shape of $\mathcal{F}_\gamma$ for various parameterizations the unknown cost function. In this subsection, we will assume that the constraint parameters θ are known, and focus only on analyzing the case of unknown γ.

B.2.1 Linear cost function parameterization

Consider the case where the cost function $c(\xi_{xu}, \gamma)$ is linear in the unknown parameters γ, i.e. $c(\xi_{xu}, \gamma) = \sum_{i=1}^{|\gamma|} \gamma_i c_i(\xi_{xu}) \doteq \gamma^\top c(\xi_{xu})$, and the constraints are fully known. The following result shows that in this setting, the set of cost function parameters consistent with the KKT conditions (3) is convex and closed under nonnegative scaling:

Theorem B.1 (Geometry of $\mathcal{F}_\gamma$ (linear in γ, known θ)). *If the cost function takes the form* $c(\xi_{xu}, \gamma) = \sum_{i=1}^{|\gamma|} \gamma_i c_i(\xi_{xu})$, *then* $\mathcal{F}_\gamma = \Gamma \cap \mathcal{C}$, *where* $\mathcal{C} \in \mathbb{R}^{|\gamma|}$ *is a convex cone in* γ-*space.*

Proof. From [29], a set $\mathcal{C}$ is a convex cone if for all γ_1, γ_2 in $\mathcal{C}$, $\alpha_1 \gamma_1 + \alpha_2 \gamma_2 \in \mathcal{C}$, for all nonnegative scalars α_1, $\alpha_2 \geq 0$. For now, assume that other than the KKT constraints, γ is unconstrained, i.e. $\Gamma = \mathbb{R}^{|\gamma|}$. Suppose we have $\gamma_1 \in \mathcal{F}_\gamma$ and $\gamma_2 \in \mathcal{F}_\gamma$. In (3), as the constraint parameters θ are known, we drop (3a), merge (3b)-(3c), and absorb $\boldsymbol{\lambda}_{\neg k}^{j\top} \nabla_{\xi_{xu}} \mathbf{g}_{\neg k}(\xi_j^{\text{loc}}, \theta)$ into the previous term. Then, if $\gamma_i \in \mathcal{F}_\gamma$ for $i \in \{1, 2\}$, we know that $\boldsymbol{\lambda}_{k,i}^j \geq \mathbf{0}$, $\boldsymbol{\lambda}_{k,i}^j \odot \mathbf{g}_k(\xi_j^{\text{loc}}) = \mathbf{0}$, and $\gamma_i^\top \nabla_{\xi_{xu}} c(\xi_j^{\text{loc}}) + \boldsymbol{\lambda}_{k,i}^{j\top} \nabla_{\xi_{xu}} \mathbf{g}_k(\xi_j^{\text{loc}}) + \boldsymbol{\nu}_{k,i}^{j\top} \nabla_{\xi_{xu}} \mathbf{h}_k(\xi_j^{\text{loc}}) = \mathbf{0}$, for $i \in \{1, 2\}$. Then if $\gamma = \alpha_1 \gamma_1 + \alpha_2 \gamma_2$, for nonnegative scalars α_1 and α_2, we can select $\boldsymbol{\lambda}_k^j = \alpha_1 \boldsymbol{\lambda}_{k,1}^j + \alpha_2 \boldsymbol{\lambda}_{k,2}^j$ and $\boldsymbol{\nu}_k^j = \alpha_1 \boldsymbol{\nu}_{k,1}^j + \alpha_2 \boldsymbol{\nu}_{k,2}^j$ to satisfy (3d); it can also be verified algebraically that this choice of $\boldsymbol{\lambda}_k^j$ satisfies the nonnegativity and complementary slackness constraints. This implies the conic hull $\mathcal{C}$ of any feasible γ is feasible for Problem 2. Finally, if $\Gamma \subset \mathbb{R}^{|\gamma|}$, we can write $\mathcal{F}_\gamma$ as the intersection of Γ and the previously constructed cone $\mathcal{C}$. $\square$

Furthermore, as Problem 2 simplifies to a linear program when θ is known, $\mathcal{F}$ is a polytope, and thus $\mathcal{F}_\gamma$ can be directly computed via a polytopic projection of $\mathcal{F}$ onto its γ coordinates [42].

B.2.2 Nonlinear cost function parameterization

For general nonlinear parameterizations of γ, the set of γ which satisfy (3) is much more challenging to represent explicitly, as checking if a γ satisfies (3) will involve satisfying a set of nonlinear, nonconvex equality constraints (3d). However, if the parameterization is a polynomial function of γ, i.e.

$c(\xi_{xu}, \gamma)$ is a polynomial in γ for fixed ξ_{xu}, $\mathcal{F}_\gamma$ can be represented as a semi-algebraic set; that is, a set described by a finite union of intersections of polynomial inequalities:

Theorem B.2. *For cost functions which are polynomial in γ for fixed ξ_{xu}, $\mathcal{F}_\gamma = \Gamma \cap \mathcal{P}$, where $\mathcal{P} \in \mathbb{R}^{|\gamma|}$ is a semi-algebraic set in γ-space.*

Proof. In this setting, $\boldsymbol{\lambda}_k^j \geq \mathbf{0}$, $\boldsymbol{\lambda}_k^j \odot \mathbf{g}_k(\xi_j^{\text{loc}}) = \mathbf{0}$, and $\nabla_{\xi_{xu}} c(\xi_j^{\text{loc}}, \gamma) + \boldsymbol{\lambda}_k^{j\top} \nabla_{\xi_{xu}} \mathbf{g}_k(\xi_j^{\text{loc}}) + \boldsymbol{\nu}_k^{j\top} \nabla_{\xi_{xu}} \mathbf{h}_k(\xi_j^{\text{loc}}) = \mathbf{0}$ define an intersection of polynomial inequalities in γ, that is, a basic semi-algebraic set. $\mathcal{F}_\gamma$ is then the projection of this set onto the γ coordinates; however, the projection of a basic semi-algebraic set may not be basic semi-algebraic in general; they are guaranteed to be semi-algebraic via the Tarski-Seidenberg theorem [43]. $\qquad \square$

While explicitly representing a semi-algebraic set can be expensive, there exist well-established methods for doing so, including exact methods like cylindrical algebraic decomposition [44] and approximate methods involving semidefinite relaxations [45].

B.3 Unknown constraints

In this section, we discuss details on the shape of $\mathcal{F}_\theta$ for unions of offset-parameterized constraints (Sec. B.3.1) and for unions of affine and higher-order parameterized constraints (Sec. B.3.2).

B.3.1 Unions of offset-parameterized constraints

Coordinate-independent parameterization:

We first analyze the case where the unknown constraint can be described as a union of offset-parameterized constraints:

Theorem B.3. *Consider the case when the unknown constraint can be described as a union of intersection of inequalities which are offset-parameterized, i.e.:*

$$\mathcal{S}(\theta) = \left\{ \kappa \in \mathcal{C} \mid \bigvee_{m=1}^{N_{ineq}} \bigwedge_{n=1}^{N_{ineq}^i} \left(g_{mn}(\kappa) \leq \theta_{mn} \right) \right\} \tag{19}$$

Then, the corresponding $\mathcal{F}_\theta$ can be described as a union of boxes as well:

$$\mathcal{F}_\theta = \left\{ \theta \in \Theta \mid \bigcup_{m=1}^{N_{box}} [I_{d \times d}, -I_{d \times d}]^\top \theta \leq s_m \right\} \tag{20}$$

Proof. In this setting, note that Problem 2 can be represented as a MILP, implying that $\mathcal{F}$ can be described as a finite union of polyhedra in the space of all decision variables. As polyhedra are closed under projection, $\mathcal{F}_\theta$ can also be represented as a finite union of polyhedra. Note that in the KKT conditions for parameterization (19), θ only appears in (3a) and (3c) (as the gradient term in (3d) drops out). Now, suppose we fix all of the boolean variables in Problem 2 (needed to implement (3a) and (3c)). Then, depending on those boolean variables for some t, j, (3c) either enforces $\theta_{mn} = g_{mn}(\kappa_t^j)$ or leaves θ_{mn} unconstrained, and (3a) imposes $\theta_{mn} \geq g_{mn}(\kappa)$ or leaves θ_{mn} unconstrained. Then, for fixed boolean variables, for any m and n, we obtain linear constraints on θ_{mn}, independent of other $\theta_{m'n'}$ for $m' \neq m$, $n' \neq n$; that is, θ_{mn} is constrained to lie in an interval. Then, unioning over all feasible boolean assignments, we obtain that $\theta_{m'n'}$ can be described as a finite union of intervals. As a result, $\mathcal{F}_\theta$ can be described as a union of boxes in θ space, as in (20). $\qquad \square$

Coordinate-dependent parameterization: If instead the constraint is parameterized such that any single constraint can depend on multiple parameters:

$$\mathcal{S}(\theta) = \left\{ \kappa \in \mathcal{C} \mid \bigvee_{m=1}^{N_{\text{ineq}}} \bigwedge_{n=1}^{N_{\text{ineq}}^i} \left(g_{mn}(\kappa) \leq \omega_{mn}^\top \theta \right) \right\} \tag{21}$$

for some fixed mixing coefficients ω_{mn}, $\mathcal{F}_\theta$ can only be represented as the more general union of polytopes, as for some m, n, the constraints on θ_{mn} will in general depend on $\theta_{m'n'}$, for $m' \neq m$, $n' \neq n$.

B.3.2 Unions of affine and higher-degree parameterized constraints

Consider the case where the constraint can be represented as:

$$\mathcal{S}(\theta) = \left\{ \kappa \in \mathcal{C} \mid \bigvee_{m=1}^{N_{\text{ineq}}} \bigwedge_{n=1}^{N_{\text{ineq}}^i} \left(g_{mn}(\kappa, \theta_{mn}) \leq 0 \right) \right\} \tag{22}$$

where $g_{mn}(\kappa, \theta_{mn})$ is affine or higher-degree in θ_{mn}. In this case, the gradient term does not drop out in (3d), meaning the set of consistent θ will depend on the projection of a set defined by polynomial equality constraints, which in general is a semialgebraic set described by polynomials of degree $2^{O(|\text{decision variables}|)}$, where "decision variables" denotes the Lagrange multipliers $\mathcal{L}$ and unknown constraint parameters θ in Problem 2.

C Obtaining a belief over constraints (expanded)

In this section, we first discuss details on zonotope extraction (Appendix C.1), how extraction can be done for the case of jointly unknown cost function and constraints (Appendix C.2), how extraction can be sped up with parallelization (Appendix C.3), and conclude with a summary of complexity and representability for the extraction problems induced by various cost and constraint parameterizations (Appendix C.4).

C.1 Other constraint parameterizations: extracting with zonotopes

While filling $\mathcal{F}_\theta$ with boxes may be efficient for a union-of-boxes constraint parameterization, infinitely many boxes may be needed to cover $\mathcal{F}_\theta$ in more general cases where $\mathcal{F}_\theta$ may only be representable as a union of polytopes or semialgebraic sets (see Appendix C for more details). To address this, we describe how to extract $\mathcal{F}_\theta$ with shapes more general than boxes while retaining efficiency. Covering $\mathcal{F}_\theta$ with polytopes instead is expensive, as polytope volume computation in high dimensions is hard. Instead, we cover $\mathcal{F}_\theta$ with zonotopes [46], which are between boxes and polytopes in representational power. A zonotope $\mathcal{Z}$ is a Minkowski sum of N_{gen} line segments: $\mathcal{Z} = \{\sum_{i=1}^{N_{\text{gen}}} \ell_i u_i \mid u_i \in [-1, 1]\}$, where $\ell_i \in \mathbb{R}^d$ is the ith segment.

Robustifying KKT to a zonotope uncertainty is done similarly to boxes: the robust constraint can be simplified with these equivalences: $a^\top(x + \sum_{i=1}^{N_{\text{gen}}} \ell_i u_i) \leq b, \forall u_i \in [-1, 1] \Leftrightarrow \max_{u_1 \in [-1,1],\ldots,u_{N_{\text{gen}}} \in [-1,1]} a^\top(x + \sum_i \ell_i u_i) \leq b \Leftrightarrow a^\top x + \sum_i |a^\top \ell_i| \leq b$. Denote (3b) and the robustified (3a), (3c), (3d) as $\text{KKT}_{\text{rob}}^{\text{zon}}(\xi_j^{\text{loc}})$.

Problem 7 (Zonotope robustification).

$$\begin{array}{ll} \underset{s,\theta,\lambda_k^j,\lambda_{-k}^j,\nu_k^j,Q_i}{\text{maximize}} & \sum_{i=1}^{N_{\text{gen}}} \|\ell_i\|_1 \\ \text{subject to} & \{\text{KKT}_{\text{rob}}^{\text{zon}}(\xi_j^{\text{loc}})\}_{j=1}^{N_{\text{dem}}} \\ & |\ell_m^\top \ell_n| \leq \delta, \ \forall m \neq n \end{array}$$

Optimizing zonotope volume is challenging, as it requires determinant computations [46] that render the overall problem a mixed integer semidefinite program, which lack reliable solvers. Instead, we optimize a surrogate, $\sum_i \|\ell_i\|_1$, and add bilinear optimization constraints to make the lines approximately orthogonal: $|\ell_m^\top \ell_n| \leq \delta$ for some small predetermined δ; these constraints are compatible with off-the-shelf solvers [34]. Finally, we cannot ignore the existential quantifiers for this parameterization, so we introduce "feedback" Lagrange multipliers. Inspired from adjustable robust optimization [47], we modify each Lagrange multiplier to take the form $\lambda_i + Q_i u$, where $Q_i u$ is a feedback term adjusting the value of the Lagrange multiplier as a linear function of the uncertainty u. The Q_i are jointly optimized to maximize the volume. The overall problem (Prob. 7) is a mixed integer bilinear program (MIBLP).

Discussion on volume maximization: Recall from the statement of Problem 7 that we are enforcing the approximate orthogonality $|\ell_m^\top \ell_n| \leq \delta$ of the line segment generators together with maximizing the line segment norms as a surrogate for volume maximization; this is to avoid the degenerate case where any two generators are parallel; this leads to the extracted volume being zero. Furthermore, we elect to use a prespecified δ instead of enforcing mutual orthogonality $\ell_m^\top \ell_n = 0$ to avoid restricting the search to rotated boxes (which is what occurs if all the generators are perfectly orthogonal).

C.2 Discussion on extracting with mixed cost function and constraint uncertainty

Extraction with mixed cost function and constraint uncertainty can be done in a similar way to Alg. 1. Specifically, we can robustify the stationarity condition to uncertainties in γ and θ jointly:

$$\nabla_{\xi_{xu}} c_\sigma(\xi_j^{\text{loc}}, \gamma + s_c \odot u_c) + \boldsymbol{\lambda}_k^{j\,\top} \nabla_{\xi_{xu}} \mathbf{g}_k(\xi_j^{\text{loc}}) + \boldsymbol{\lambda}_{\neg k}^{j\,\top} \nabla_{\xi_{xu}} \mathbf{g}_{\neg k}(\xi_j^{\text{loc}}, \theta + s \odot u) + \boldsymbol{\nu}_k^{j\,\top} \nabla_{\xi_{xu}} \mathbf{h}_k(\xi_j^{\text{loc}}) = \mathbf{0}$$
(23)

The remaining KKT conditions are unchanged, as γ does not factor into the other constraints. The modified stationarity condition can be robustified in a similar way. We denote (3b), the robustified (3a) and (3c), and the joint cost/constraint-robustified (3d) together as $\text{KKT}_{\text{rob,cost}}^{\text{box}}(\xi_j^{\text{loc}})$. We can then modify Problem 4 to account for the additional scaling variables as such:

$$\begin{aligned}
\underset{s, s_c, \theta, \gamma, \mathcal{L}}{\text{maximize}} \quad & \left(\textstyle\prod_i s_i \prod_i s_{c_i} \right)^{1/d} \\
\text{subject to} \quad & \{ \text{KKT}_{\text{rob,cost}}^{\text{box}}(\xi_j^{\text{loc}}) \}_{j=1}^{N_{\text{dem}}}
\end{aligned}$$

This new optimization problem can be integrated into Algorithm 1, repeatedly carving out subsets of $\mathcal{F}_\theta \times \mathcal{F}_\gamma$.

C.3 Speeding up extraction with parallelization

We sketch one possible way that Alg. 1 can be sped up with parallelization on M cores:

- Partition the parameter space Θ into M disjoint boxes.
- Run Alg. 1 on each partition separately.
- Reconstruct $\mathcal{F}_\theta$ by unioning the extracted parameters from each partition.

C.4 Summary on problem complexity

In the following table, we organize the representability of particular constraint learning and feasible set extraction problems, for various constraint parameterizations. To summarize, the case of unknown constraints induces a set of integer variables due to the complementary slackness condition. The remaining complexity depends on the complexity of the constraint and cost function parameterizations. Furthermore, the extraction problems are only conic-representable due to our formulation of volume maximization.

Constraint parameterization	Cost parameterization	Problem 2, class	Class (extraction)
Known	Linear	LP	Direct projection
Union of offsets	Known	MILP	MISOCP
Union of offsets	Linear	MILP	MISOCP
Union of affine	Known	MIBLP	MIBLP
Union of affine	Linear	MIBLP	MIBLP
Nonlinear	Nonlinear	MINLP	MINLP

D Policies for adaptive constraint satisfaction (expanded)

In this appendix, we first discuss fast reformulations for the chance-constrained planning problem (Appendix D.1), expanded details on the sampling-based planners that we use when the optimization constraints are not MICP-representable (Appendix D.2), discussion on extending Problem 6 to handle priors other than the uniform distribution (Appendix D.3), and discussion on how to perform belief updates for various constraint sensing modalities (Appendix D.4).

D.1 Fast reformulations of Problem 6

As written, Problem 6 can be expensive to solve due to the many possible assignments of the box decision variables b_i^{cen} and b_i^{scale}, for each i, and the combinatorial coupling between boxes $i \neq j$. Furthermore, the non-convex norm constraint (11f) causes Problem 6 to be an MIBLP, which are

in general more challenging to solve than mixed integer convex programs. This can cause solving Problem 6 to be slow. To address this, we propose two reformulations:

- In the first, we still optimize over the N_{box} boxes simultaneously, and simply replace the non-convex constraint (11f) with a linear approximation: $\sum_i t_i \geq 1 - \varepsilon$. However, by doing this, the original chance-constraint $\Pr(\xi_{xu} \text{ safe}) \geq 1 - \varepsilon$ may not hold exactly. One can get around this by instead enforcing $\sum_i t_i \geq 1 - \tilde{\varepsilon}$, incrementally shrinking $\tilde{\varepsilon}$ until the resulting trajectory satisfies the original chance constraint, but this can be cumbersome.

- The second reformulation, which is what we use in practice to solve Problem 6-ε_{min}, instead optimizes over the boxes one at a time. That is, if we are given a budget of N_{box} boxes, we solve Problem 6-ε_{min} for $N_{\text{box}} = 1$, then at the next iteration, solve for $N_{\text{box}} = 2$, where the first box is fixed. This continues until we reach the box budget. If N_{box} is chosen to be large enough that the covered probability mass is the same when solving Problem 6 for N_{box} and $N_{\text{box}} + 1$, this sequential strategy is lossless, i.e. will return the same solution as Problem 6 without relaxations, for the case where choosing to satisfy one possible constraint can never cause the trajectory to not be able to satisfy a different constraint. In other cases, this strategy may lead to convergence to a local optimum in the amount of probability mass covered, but we observe that this strategy works well in practice.

D.2 Sampling-based planners

We utilize two sampling-based planners to compute open-loop plans in the case where the constraints of Problem 6 are not representable in a mixed integer convex program: Minimum Constraint Removal (MCR) and the Blindfolded Traveler's Problem (BTP).

Minimum Constraint Removal (MCR) [35]: MCR takes a *finite* set of constraints and a start/goal state. At each step of the algorithm, MCR keeps track of the path from the start to the goal that violates the least number of constraints so far. The algorithm initializes this with the straight-line edge between the start and goal states, and sets k, the minimum number of constraints that must be violated as the number of constraints that the start/goal state violate. Then, MCR incrementally grows a roadmap, where candidate expansions are limited based on the number of constraints the candidate edge will violate, and at each growth iteration, finds the path from the start to each vertex which violates the minimum number of constraints and updates the path to the goal which violates the least constraints. Every so often, we increase k to allow for more constraints to be violated when expanding the roadmap. This is summarized in Section 4.2 of [35].

To use MCR to approximate Problem 6-ε_{max}, we sample N_{sample} constraints from the belief $\{\theta_i \sim b(\theta)\}_{i=1}^{N_{\text{sam}}}$ as input to MCR, then run MCR as just described. The output of MCR will then be a path on the roadmap connecting the start and goal which violates the minimal number of *sampled constraints*.

The Blindfolded Traveler's Problem (BTP) [36]: The Blindfolded Traveler's Problem (BTP) can be modeled as a graph search problem. It is defined by a graph $G = (V, E, W, x_0, x_g)$ with vertices V, edges E, weights C, and start/goal state x_0 and x_g. Furthermore, each edge e is invalid with probability $p(e) \in [0, 1]$. If an edge e_{uv} from u to v is traversed, either the traversal is successful, and the agent ends up at vertex v, or the agent discovers that the edge is invalid $\eta_{uv} \in [0, 1]$ fraction of the way through, at which point the agent needs to turn back and return to vertex u; such a traversal attempt costs $2\eta_{uv} C_{uv}$, where $c_\sigma(e_{uv})$ is the cost of traversing edge uv. The agent has a prior over the probability of edge validity and blockage, which can be updated based on the observations gained through traversing the graph. A solution to BTP is a policy which takes the start state, history of observations, and outputs an edge to traverse.

While computing an optimal policy for the BTP is NP-complete, it is possible to compute high-quality approximations, for example using the Collision Measure strategy (Section 5.1 of [36]). This strategy approximates BTP by modifying the graph edge weights to penalize the log probability of the edges being unsafe, that is, edge weights are modified such that $\tilde{c}_{uv} = c_{uv} - \beta \log(p(e_{uv} \text{ safe})$, for some weights β. We then compute paths on the graph by running A* with the modified edge weights.

Specifically, to approximate Problem 6-R with BTP, we sample constraints $\{\theta_i \sim b(\theta)\}_{i=1}^{N_{\text{sam}}}$ from the belief $b(\theta)$ to approximate the probabilities $p(e_{uv} \text{ safe})$; specifically, for each edge, we estimate $p(e_{uv} \text{ safe})$ as the fraction of sampled constraints which are violated.

D.3 Priors $p(\theta)$ other than the uniform distribution

In Section 5.1, we discuss how to integrate over a *uniform prior* to optimize boxes over the probability density which can be embedded in an MISOCP for planning probabilistically safe trajectories. Here, we discuss a different prior which also satisfies the closed-form integrability and log-concave assumptions detailed in Sec. 5.1: $p(\theta) \propto \prod_{i=1}^{|\kappa|}(\bar{\kappa}_i(\theta) - \underline{\kappa}_i(\theta))$, where $\bar{\kappa}_i$, $\underline{\kappa}_i$ denote the upper and lower bounds of the box in dimension i of the constraint space. This prior places more probability mass on larger box constraints; hence, the behavior generated using this prior is more conservative. As a concrete example, see Fig. 9 for trajectories solving Problem 6 for the different priors, solved over a range of different start/goal states. Observe that the trajectories that use the weighted prior are more conservative. Generally, an investigation of other useful priors which satisfy our assumption of closed-form integrability is the subject of future work.
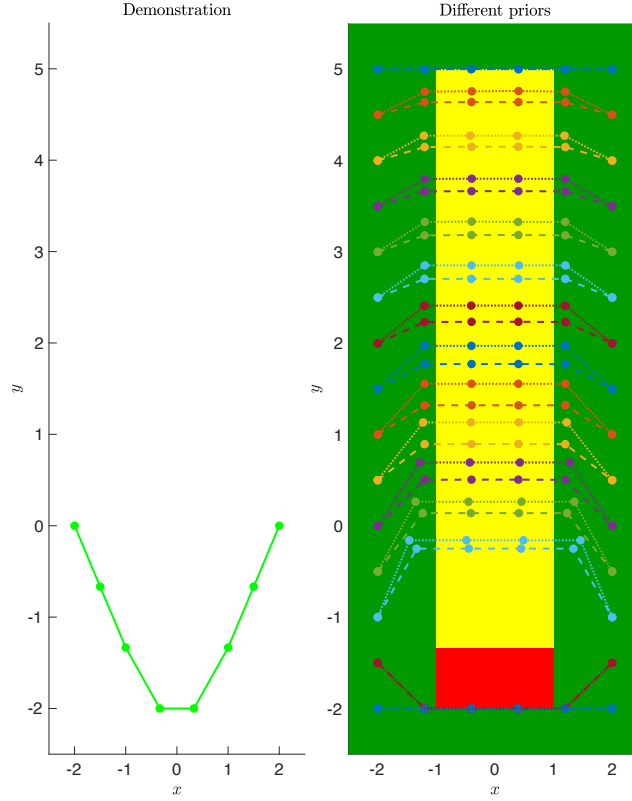

Fig. 9: Trajectories generated for different priors. We are provided one demonstration (left), which reveals the left, right, and bottom extents of a box obstacle constraint, but not the upper extent. Dashed lines correspond to the uniform prior $p(\theta) \propto 1$, while dotted lines correspond to the prior $p(\theta) \propto \prod_{i=1}^{|\kappa|}(\bar{\kappa}_i(\theta) - \underline{\kappa}_i(\theta))$.

D.4 Belief updates

Given an initial set of consistent constraint parameters $\mathcal{F}_\theta$, we are interested in updating $\mathcal{F}_\theta$ to be consistent with constraint information gathered in execution. We specifically write belief updates for the following constraint sensing modalities:

Direct, exact measurements: Consider a measurement that κ_{safe} is safe, given an initial set of consistent constraints $\mathcal{F}_\theta$. We want to find the maximum volume subset of $\mathcal{F}_\theta$ which satisfies $g(\kappa, \theta) \leq 0$. To accomplish this, we simply modify Problem 4 to:

$$\underset{s}{\text{maximize}} \quad \left(\textstyle\prod_i s_i\right)^{1/d}$$
$$\text{subject to} \quad g(\kappa_{\text{safe}}, \theta + s \odot u) \le 0$$

where we can eliminate the uncertain variable u with the identity used in Section 4.1, and use this modified problem in Algorithm 1. Note that as the local optimality of the demonstrations is already embedded in the initial $\mathcal{F}_\theta$, we do not need to add them as additional constraints in this modified problem, improving the computation time.

The same modification can be done an unsafe measurement κ_{unsafe}, except with a constraint $g(\kappa_{\text{unsafe}}, \theta + s \odot u) > 0$.

Ranged, exact measurements: This case can come up when given LiDAR scans of the environment obtained in execution. For this setting, we assume that we are given a finite set of states which are all sensed to be safe, or all sensed to be unsafe. In our examples, we obtain this finite set of points by discretizing the possibly continuous ranged LiDAR measurement using a grid of measurement locations. This is a simple extension of the previously discussed modification for a single observed state; we simply have to add constraints corresponding to each state, and use this modified problem in Algorithm 1:

$$\underset{s}{\text{maximize}} \quad \left(\textstyle\prod_i s_i\right)^{1/d}$$
$$\text{subject to} \quad g(\kappa_{\text{safe}}^i, \theta + s \odot u) \le 0, \quad i = 1, \dots, N_{\text{safe}}$$
$$\qquad\qquad g(\kappa_{\text{unsafe}}^i, \theta + s \odot u) > 0, \quad i = 1, \dots, N_{\text{unsafe}}$$

Direct, uncertain measurements: In this case, suppose that we are given an initial set of consistent constraints $\mathcal{F}_\theta$ as well as a finite set of states which may be possibly unsafe (the same ideas extend to the case where a set of states may be possibly safe); that is, we are given $\mathfrak{C}_{\neg s}$, where we learn in execution that at least one element of $\mathfrak{C}_{\neg s}$ is unsafe: $\exists \mathfrak{C}_{\neg s}^i \in \mathcal{A}(\theta^*)$. In our examples for the 7-DOF arm, we obtain this finite set of points by discretizing the continuous set of points which could be in contact by sampling points on the surface of the arm on the links downstream from where a torque limit is violated. Again, a similar modification can be made to Problem 4:

$$\underset{s}{\text{maximize}} \quad \left(\textstyle\prod_i s_i\right)^{1/d}$$
$$\text{subject to} \quad \bigvee_{i=1}^{|\mathfrak{C}_{\neg s}|} g(\mathfrak{C}_{\neg s}^i, \theta + s \odot u) \le 0$$

Specifically, the logical constraints over which state is unsafe can be modeled with binary variables, so the overall problem is still an MISOCP. The modified problem can be used in Algorithm 1.

E Theory

In this appendix, we provide proofs for the theorems in the main body of the paper.

Theorem E.1. *If Alg. 1 terminates for any parameterization, its output is guaranteed to cover $\mathcal{F}_\theta$.*

Proof. Suppose for contradiction that Algorithm 1 terminates such that $\mathcal{F}_\theta \setminus \left(\bigcup_{i=1}^{N_{\text{infeas}}} \hat{\mathcal{F}}_\theta^i\right) = \mathcal{F}_\theta^{\text{remain}} \ne \emptyset$. However, by construction, Alg. 1 only terminates if there does not exist any $\theta \in \Theta$ for which $\{\text{KKT}(\xi_j^{\text{loc}})\}_{j=1}^{N_{\text{dem}}}$ can be satisfied; otherwise, Prob. 4 remains feasible. For all $\theta \in \mathcal{F}_\theta^{\text{remain}}$, by definition of being an element of $\mathcal{F}_\theta$, there exist $\mathcal{L}$ to satisfy $\{\text{KKT}(\xi_j^{\text{loc}})\}_{j=1}^{N_{\text{dem}}}$. Contradiction. $\square$

Theorem E.2. *Alg. 1 is guaranteed to terminate in finite time for union-of-boxes parameterizations.*

Proof. From Theorem B.3, $\mathcal{F}_\theta$ can be described as a union of a finite number of axis-aligned rectangles: $\mathcal{F}_\theta = \bigcup_{i=1}^{N_{\text{box}}} \mathcal{B}_i\}$. Extend each hyperplane defining the boundary of a box $\mathcal{B}_i$ to infinity to obtain an irregular grid $\{\mathcal{G}_i\}_{i=1}^{N_{\text{grid}}}$ over Θ (see Figure 10 for the case in 2D). As $\mathcal{F}_\theta$ is composed of a
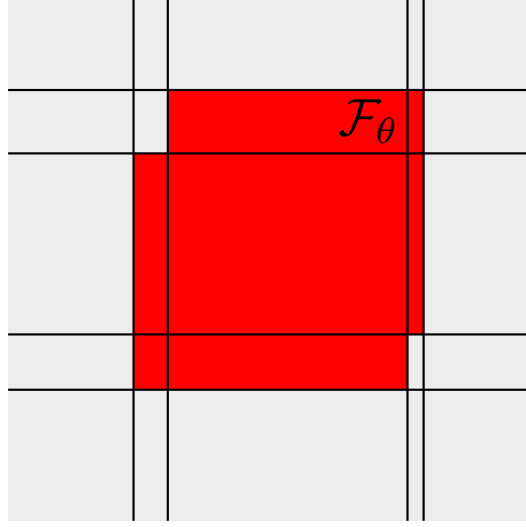
Fig. 10: Grid used in the proof of Theorem E.2.

finite number of boxes and hence there are a finite number of extended hyperplanes, there will be a finite number of grid cells, i.e. N_{grid} is finite.

We now prove that the solution of Problem 4 at any iteration i, $\hat{\mathcal{F}}_\theta^i$, can be exactly represented by some subset of grid cells: $\hat{\mathcal{F}}_\theta^i = \{\kappa \mid [I_{d \times d}, -I_{d \times d}]^\top \kappa \leq [\bar{\theta}, -\underline{\theta}]^\top\} = \bigcup_{j=1}^{N_{\text{rep}}} \mathcal{G}_j$. Suppose for contradiction that there exists some grid cell $\mathcal{G}_k$ that $\hat{\mathcal{F}}_\theta^i$ only partially contains: $(\mathcal{G}_k \cap \hat{\mathcal{F}}_\theta^i \neq \emptyset) \wedge (\mathcal{G}_k \cap \hat{\mathcal{F}}_\theta^i \neq \mathcal{G}_k)$. Formally, this means that in some coordinate of θ, say the mth coordinate, the upper bound of $\hat{\mathcal{F}}_\theta^i$, $\bar{\theta}_m^i$ satisfies $\bar{\theta}_m^i \in [\underline{\theta}_m^k, \bar{\theta}_m^k]$, where these denote the lower and upper bounds of grid k in dimension m; similar logic holds for analyzing the lower bound. For $\bar{\theta}_m^i$ to be the upper bound of $\hat{\mathcal{F}}_\theta^i$ in the mth coordinate, by the optimality of Problem 4, there must exist some constraint state κ contained in the expanded box $\{\kappa \mid [I_{d \times d}, -I_{d \times d}]^\top \kappa \leq [\bar{\theta}_1, \ldots, \bar{\theta}_{m-1}, \bar{\theta}_m^k, \bar{\theta}_{m+1}, \ldots, \bar{\theta}_d, -\underline{\theta}]^\top\}$ such that $\kappa \notin \mathcal{F}_\theta$. However, this is not possible, as by the grid partition, there exists no hyperplane defining $\mathcal{F}_\theta$ that can be crossed in the mth coordinate between $\underline{\theta}_m^k$ and $\bar{\theta}_m^k$. Contradiction.

Finally, as each iteration in Alg. 1 removes a finite number of grid cells, Alg. 1 will terminate in a finite number of iterations. $\square$

Theorem E.3. *A solution to Prob. 6 is a guaranteed feasible, possibly suboptimal solution to Prob. 5.*

Proof. Feasibility follows by construction of Problem 6, as constraint (11f) directly models the probability constraint (10d): $\int_{\mathcal{B}_i} d\theta = t_i^d$, so $\sum_i t_i^d = \sum_i \int_{\mathcal{B}_i} d\theta = \int_{\Theta_s} d\theta$ for $\Theta_s = \bigcup_{i=1}^{N_{\text{box}}} \mathcal{B}_i$, which is exactly $\Pr(\xi_{xu} \text{ safe})$ when integrated over Θ_s, for the uniform prior $b_{\text{dem}}(\theta)$.

Suboptimality arises from the optimal partition of probability possibly not being representable as a union of boxes: in general, there exists Θ_s under which $c_\sigma(\xi_{xu})$ is minimized, such that there does not exist N_{box} boxes where $\Theta_s = \bigcup_{i=1}^{N_{\text{box}}} \mathcal{B}_i$. $\square$

F   Further experimental details

In this section, we provide additional details on our experimental results. We first discuss in detail the baseline algorithms that we compare to in the results (Appendix F.1). We then demonstrate closed-loop planning with an uncertain nonlinear constraint using a sampled approximation of Problem 6 (Appendix F.2), and then discuss additional details and visualize example runs of our method and baseline approaches for the mixed quadrotor uncertainty example (Appendix F.3), the 7-DOF arm example (Appendix F.4), and the quadrotor maze example (Appendix F.5).

F.1 Planning baselines

F.1.1 Mixed quadrotor example

Scenario approach: The scenario approach [26] satisfies uncertain constraints by sampling N_{sam} possible constraint parameters $\{\theta_i\}_{i=1}^{N_{\text{sam}}}$ and finds a solution that satisfies all of the sampled constraints. For this example, we may not be able to satisfy all of the sampled constraints, and hence the scenario approach may render the problem infeasible. To get around this to use the method as a baseline, we iteratively sample additional constraints, and solve the following open-loop planning problem:

$$
\begin{aligned}
\min_{\xi_{xu}} \quad & c_\sigma(\xi_{xu}) \\
\text{s.t.} \quad & \bar{\phi}(\xi_{xu}) \in \bar{\mathcal{S}} \subseteq \bar{\mathcal{C}}, \ \phi_\sigma(\xi_{xu}) \in \mathcal{S}_\sigma \subseteq \mathcal{C}_\sigma \\
& \xi_{xu} \in \mathcal{S}(\theta), \ \forall \theta \in \{\theta_i\}_{i=1}^{N_{\text{sam}}}
\end{aligned}
$$

We stop sampling constraints when the problem becomes infeasible and return the feasible trajectory generated at the previous iteration.

Optimistic approach: In this approach, we are optimistic about the true constraint, only avoiding the set of guaranteed-unsafe states, buffering the extents by 0.5 in the uncertain constraint dimensions.

F.1.2 7-DOF arm example

BTP without constraint parameterization or demonstrations: In this version of BTP, we provide neither a union-of-boxes constraint parameterization nor a set of demonstrations. Instead, collision probabilities are measured with "Collision Hypothesis Sets" (CHS) [38], which use a voxelization of the environment, with probabilities of particular voxels being occupied updated based on the occupancy of the robot volume during collision.

Optimistic approach: In this approach, we use the same graph provided to BTP and do not provide the information provided by the demonstrations, and we iteratively solve an optimistic problem. At the first iteration, we find run A* with all edges on the graph assumed valid, and attempt to execute the path. If the robot collides when traversing some edge e_{uv} from vertex u to v, the robot backtracks to vertex u, removes edge e_{uv} from the graph, and replans on the modified graph. The procedure continues until the robot is at the goal.

F.1.3 Quadrotor maze example

Guaranteed-safe planning: In this approach [1], we compute paths which are guaranteed-safe with respect to the constraint uncertainty. Under the assumption that the constraint parameterization is correct, this guarantees that we will never need to replan upon discovering a constraint, but at the cost of possibly high-cost, conservative trajectories.

Optimistic approach: This approach [27] is optimistic with respect to the uncertain space, and constructs high-level plans that plan to intermediate goals on the frontier of unknown space, between the current state and the goal, and executes the best high-level plan. In more detail, we replicate the approximations used in Section IV.B of [27]. In particular, we assume that the unknown space is free, but buffer known obstacles by 0.1 meters in the uncertain dimensions. We discretize the unknown frontier in 2 meter intervals to construct our subgoals.

F.2 Nonlinear constraint

We show that our method can plan with constraint beliefs for non-union-of-boxes constraint parameterizations. Specifically, we are given a demonstration on a 2D kinematic system $[\chi_{t+1}, y_{t+1}]^\top = [\chi_t, y_t]^\top + [u_t^\chi, u_t^y]^\top$ which minimizes path length $c(\xi) = \sum_{t=1}^{T-1} \|x_{t+1} - x_t\|^2$ (Figure 11.A) while satisfying the constraint $g(x, \theta) = \theta_1(x_1^4 + x_2^4) + \theta_2(x_1^3 + x_2^3) + \theta_3(x_1 - 1)^3 + \theta_4(x_2 + 1)^3 > 2$ for $\theta = [2, -5, 5, 5]^\top$. As the demonstration is rather uninformative, many θ make it locally-optimal. As the constraint is affine in θ, we extract $\mathcal{F}_\theta$ with zonotopes (running Algorithm 1 with Problem

7); see Figure 11.B-C for $\mathrm{proj}_\mathcal{X}(\mathcal{F}_\theta)$ and a corresponding probability heatmap. We now want to solve Prob. MCV, planning from $x_0 = [0, 0.75]^\top$ to $x_g = [0, -1.5]^\top$. As $g(x, \theta)$ is not MICP-representable, we solve an approximation of Prob. MCV with samples, sampling 100 constraints from $b_{\mathrm{dem}}(\theta)$ as input to MCR (c.f. Section 5.2). The resulting path, Plan 1, (Figure 11.C) violates one possible constraint at $x_{\neg s} = [0.49, 0.8]^\top$. Though this path is safe for the true constraint, this is unknown to the learner, so we also precompute a contingency. Updating the belief $b_{\mathrm{ex}}(\theta)$ with $\mathfrak{C}_{\neg s} = \{x_{\neg s}\}$ and previously visited states as $\mathfrak{C}_s$ reduces the constraint uncertainty (Figure 11.D-E). In particular, we note that the measurement at $[0.49, 0.8]^\top$ also removes the constraint uncertainty on the left-hand side of the space; this is due to the nonlinearity of the constraint parameterization, and as we can see in the belief (Figure 11.C), only some of the the convex obstacles that cover the right side of the space can explain the possible collision at $[0.49, 0.8]^\top$. Hence, this measurement will update the belief to eliminate the non-convex obstacles, removing the left-side uncertainty. For this updated belief, Contingency 1 can be planned with no constraint violations (Figure 11.E). Extracting $\mathcal{F}_\theta$ with Algorithm 1 and planning with MCR takes 30 min. (can be sped up with a parallel implementation, c.f. Appendix C) and 30 sec., respectively.
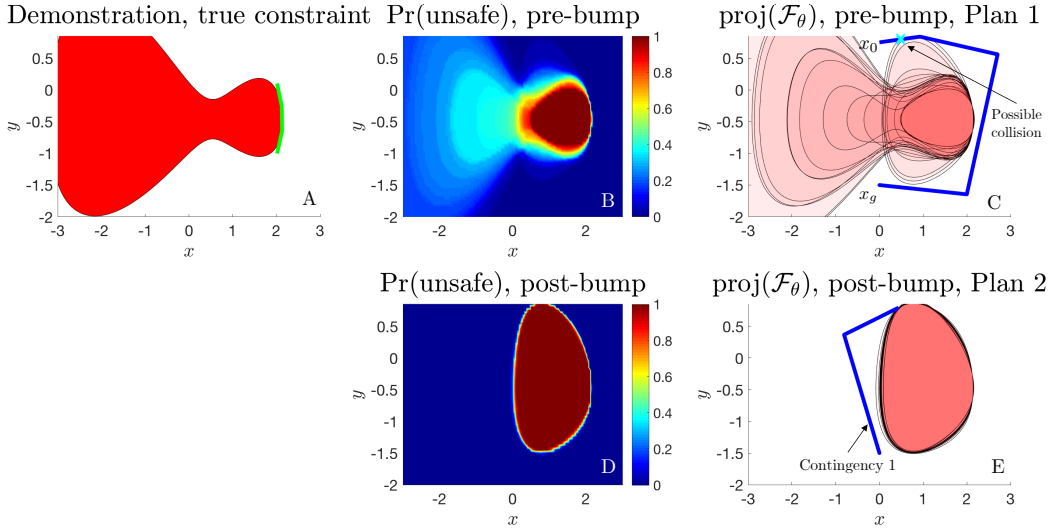

Fig. 11: Example demonstrating MCR on planning with a nonlinear constraint. **A**: Demonstration, overlaid with the true constraint (red). **B**: Initial constraint uncertainty, visualized as a normalized probability heatmap.**C**: The initial plan (blue) generated by MCR, overlaid by a subsampling of 20 of the sampled constraint parameters θ provided to MCR. A possible collision occurs at the cyan "x". **D**: The updated constraint uncertainty probability heatmap were the cyan state to be in collision. **E**: The new plan for the updated constraint uncertainty reaches the goal without violating any possible sampled constraints.

F.3 Mixed state-control constraint uncertainty on a quadrotor

The system dynamics for the quadrotor [48] are:

$$
\begin{bmatrix} \dot{\chi} \\ \dot{y} \\ \dot{z} \\ \dot{\alpha} \\ \dot{\beta} \\ \dot{\gamma} \\ \ddot{\chi} \\ \ddot{y} \\ \ddot{z} \\ \ddot{\alpha} \\ \ddot{\beta} \\ \ddot{\gamma} \end{bmatrix} = \begin{bmatrix} \dot{\chi} \\ \dot{y} \\ \dot{z} \\ \dot{\beta}\frac{\sin(\gamma)}{\cos(\beta)} + \dot{\gamma}\frac{\cos(\gamma)}{\cos(\beta)} \\ \dot{\beta}\cos(\gamma) - \dot{\gamma}\sin(\gamma) \\ \dot{\alpha} + \dot{\beta}\sin(\gamma)\tan(\beta) + \dot{\gamma}\cos(\gamma)\tan(\beta) \\ -\frac{1}{m}[\sin(\gamma)\sin(\alpha) + \cos(\gamma)\cos(\alpha)\sin(\beta)]u_1 \\ -\frac{1}{m}[\cos(\alpha)\sin(\gamma) - \cos(\gamma)\sin(\alpha)\sin(\beta)]u_1 \\ g - \frac{1}{m}[\cos(\gamma)\cos(\beta)]u_1 \\ \frac{I_y - I_z}{I_x}\dot{\beta}\dot{\gamma} + \frac{1}{I_x}u_2 \\ \frac{I_z - I_x}{I_y}\dot{\alpha}\dot{\gamma} + \frac{1}{I_y}u_3 \\ \frac{I_x - I_y}{I_z}\dot{\alpha}\dot{\beta} + \frac{1}{I_z}u_4 \end{bmatrix}, \tag{25}
$$

24

We time-discretize the dynamics by performing forward Euler integration with discretization time $\delta t = 0.7$. The 12D state is $x = [\chi, y, z, \alpha, \beta, \gamma, \dot{x}, \dot{y}, \dot{z}, \dot{\alpha}, \dot{\beta}, \dot{\gamma}]^\top$, and the relevant constants are $g = -9.81\text{m/s}^2$, $m = 1\text{kg}$, $I_x = 0.5\text{kg} \cdot \text{m}^2$, $I_y = 0.1\text{kg} \cdot \text{m}^2$, and $I_z = 0.3\text{kg} \cdot \text{m}^2$.

The simplified double integrator model that we use to plan in Problem 6 is as follows:

$$
\begin{bmatrix} \dot{\chi} \\ \dot{y} \\ \dot{z} \\ \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \chi \\ y \\ z \\ \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ g \end{bmatrix}
\tag{26}
$$

which is then time discretized with $\delta t = 0.5$, with $g = -9.81\text{m/}s^2$.
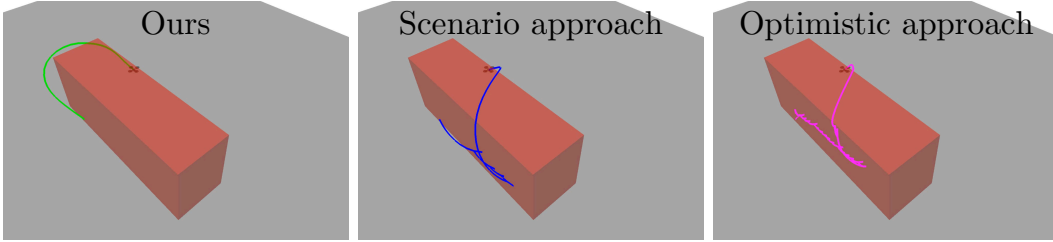
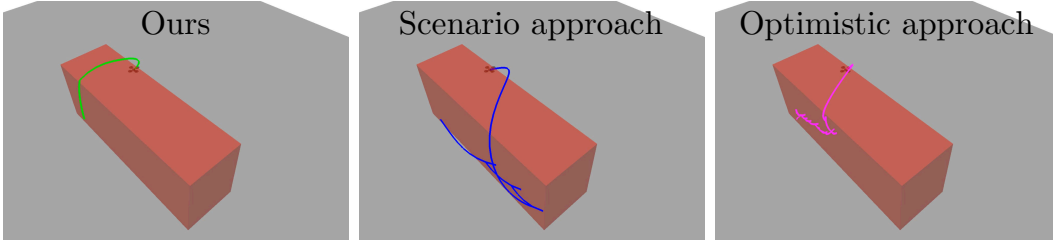Fig. 12: Example run 1 (mixed quadrotor example).

Fig. 13: Example run 2 (mixed quadrotor example).

Visualizing example runs: For two different sampled ground-truth constraints, we visualize the trajectories executed by our policy, the scenario approach policy [26], and the optimistic policy to compare their properties.

In Figure 12, we display the sampled state constraint in red, and the sampled control constraint is $\|u\|_2^2 \leq 98.27$. Our policy, which attempts to satisfy all of the state constraints by trying to move above all of the possible obstacles (see Sec. 6 for more discussion), violates the control constraint at the first time-step by attempting to do so. Our policy then switches to the first contingency plan and successfully reaches the goal with 1 constraint violation. On the other hand, the scenario approach suffers 6 constraint violations (due to sampling constraints), while the optimistic approach suffers 25 constraint violations (because it does not try to avoid any states other than those which are known to be unsafe).

In Figure 13, we display the sampled state constraint in red, and the sampled control constraint is $\|u\|_2^2 \leq 98.55$. Our policy reaches the goal in one try (0 constraint violations), as we do not end up violating the control constraint. On the other hand, the scenario approach suffers 4 constraint violations, while the optimistic approach suffers 15 constraint violations.

Histogram: In computing Plan 1 (the initial executed trajectory) and Contingencies 1-4 (the trajectories we switched to upon observing 1, 2, 3, and 4 constraint violations), we can calculate the volumes of the covered $\mathcal{B}_i$ which are optimized by Problem 6 at each iteration. When computing Plan 1, we cover $p_0 = 69.10\%$ of the possible constraints; when computing Contingency 1, we cover $p_1 = 54.90\%$ of the possible constraints under the updated belief, $p_2 = 56.33\%$ for

25

Contingency 2, $p_3 = 73.91\%$ for Contingency 3, and $p_4 = 98.00\%$ for Contingency 4. Using these percentages, we can calculate the theoretical frequency of overrides as $p(0 \text{ overrides}) = p_0$, $p(1 \text{ override}) = (1 - p_0)p_1$, and so on, until $p(4 \text{ overrides}) = \prod_{i=1}^{3}(1 - p_i)p_4$. Using these formulae, we compute the theoretical probability of suffering i constraint violations before reaching the goal, which we compare with the empirical histograms (normalized over 500000 trials) (Fig. 14), and we see the statistics match quite closely.
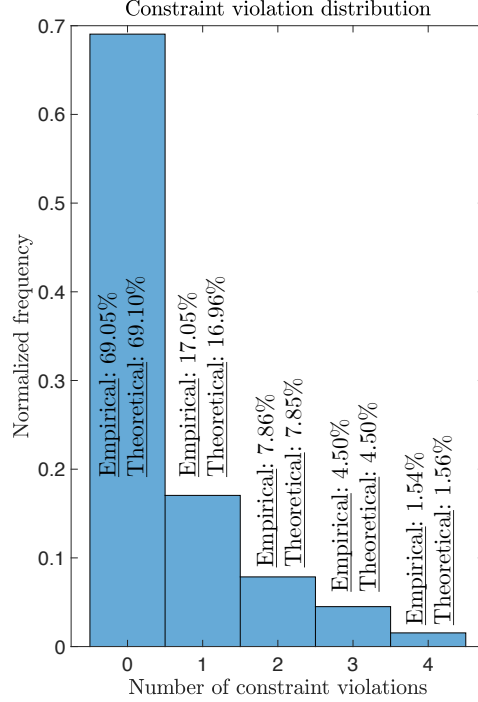
Fig. 14: Constraint violation histogram for the mixed quadrotor uncertainty example.

F.4   7-DOF arm with contact sensing uncertainty

We use a kinematic model of the arm: $x_{t+1}^i = x_t^i + u_t^i$, for $i = 1, \ldots, 7$. Here, $x \in \mathbb{R}^7$, where coordinate i of the state denotes the angle of the ith joint. For planning, we use a BTP graph with 5000 vertices (c.f. Section 5.2). In the following, we discuss more details on the performance of different policies on the task discussed in Section 6.

BTP with CHS: We present a time-lapse of the trajectory executed by running BTP with CHS (as described in Appendix F.1.2) in Figure 15. Since this approach is not given demonstrations, it requires several bumps in order to sufficiently localize the shelf. Furthermore, this approach does not leverage a union-of-boxes constraint parameterization as a prior on the world, as the CHS does not extrapolate about the constraint beyond the sensed collision, making it so that more bumps occur before reaching the goal.

Optimistic approach: We present the trajectory executed by running the optimistic strategy described in Appendix F.1.2 in Figure 16. Since this strategy entirely ignores the correlation in validity between edges which are close to each other, it explores many edges, yielding a trajectory cost which is much higher than the other approaches.

Suboptimal human demonstrations: Finally, we present a time-lapse of our policy when initialized with suboptimal human demonstrations in Figure 18. The demonstrations are captured using an HTC Vive in a virtual reality simulation environment (Figure 17). Overall, the behavior of our policy when initialized with these human demonstrations is similar to the the case of synthetic demonstrations (which is discussed in Sec. 6): it plans to move around the shelf, and in doing so, collides with the unmodeled obstacle. This triggers a constraint parameterization update, and the replanned path (which avoids the shelf and the uncertain region induced by the collision) steers the arm to the goal without further collision. For this case, the executed trajectory cost is 7.78 rad.
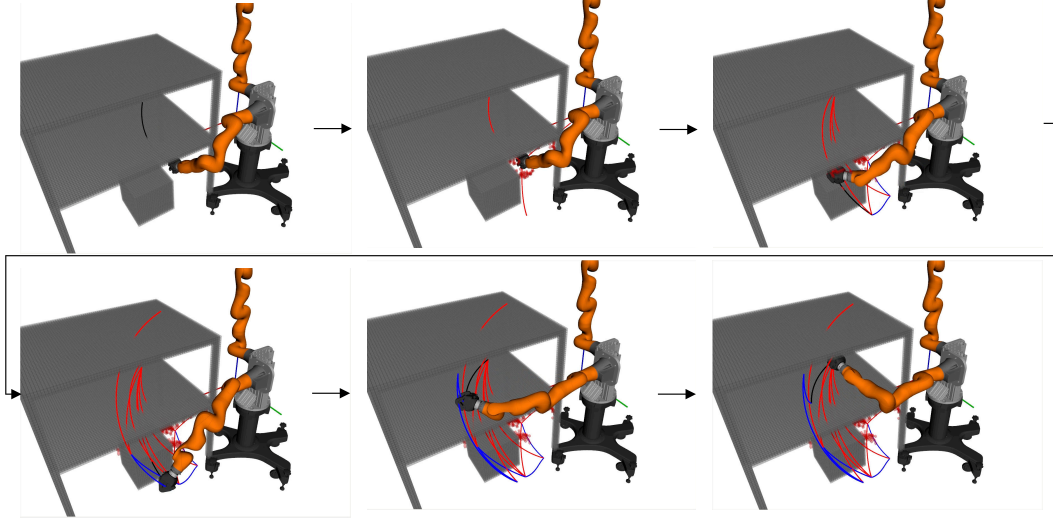
Fig. 15: BTP with CHS. Voxels are colored red if they are possibly unsafe according to the CHS. Red edges are attempted edges which are discovered to be blocked in execution. The attempted edges which were unblocked are colored blue.
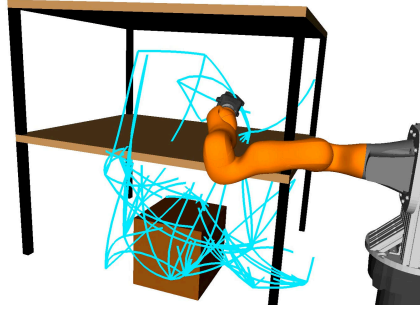

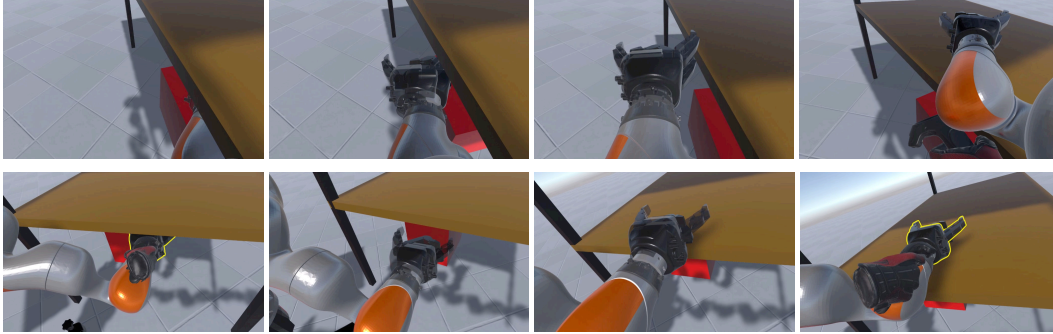Fig. 16: Trajectory executed by the optimistic policy.

Fig. 17: Suboptimal human demonstrations. Top row: time-lapse of the first demonstration. Bottom row: time-lapse of the second demonstration.

F.5 Quadrotor maze

Inevitable collision states: We integrate the inevitable collision state [39] avoidance constraints into our approach. We first enforce that on a planned trajectory, each state which may be possibly unsafe must be observable (within 2 meters) from a previous state on the trajectory. Furthermore, the line of sight between this previous state and the possible unsafe state cannot be occluded by any obstacle other than the obstacle which is making the state possibly unsafe. For these possibly unsafe states, we enforce that we can brake in time to avoid collision (and together with the line-of-sight
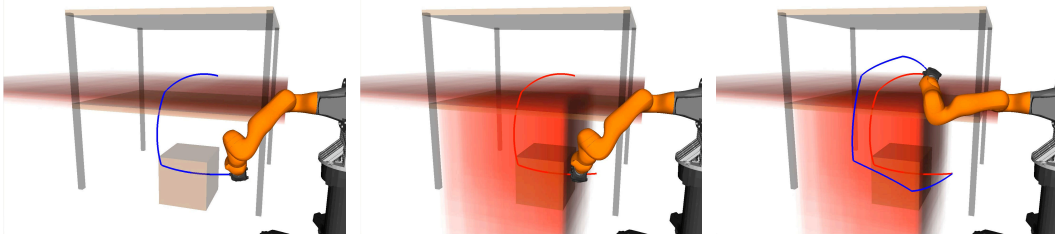
27

Fig. 18: Policy when initialized with suboptimal demonstrations. Left: plans an initial trajectory that bumps into the unmodeled obstacle. Center: constraint parameterization is updated to two boxes. Right: replanned trajectory successfully avoids all collisions, steering the arm to the goal.

constraint, enforces that we can also sense if we need to brake). This is done by explicitly optimizing "brake trajectories" in conjunction with the planned trajectory, which are rooted two time-steps on the plan before a possible collision and which bring the system to a stop without violating any possible constraints. We further enforce the line of sight constraint by discretizing the line segment between x_t and x_{t+2} (if x_{t+2} is possibly unsafe) into 10 points, and enforcing that each discretized state is guaranteed to satisfy all constraints other than the uncertain constraint which can make x_{t+2} possibly unsafe.
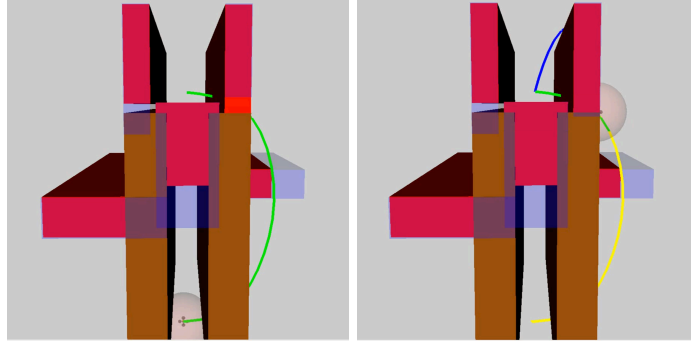

Fig. 19: Example run, our policy (quadrotor maze). Initial plan (green), contingency plan (blue), actually executed plan (yellow). The sphere around the quadrotor indicates the sensing radius.
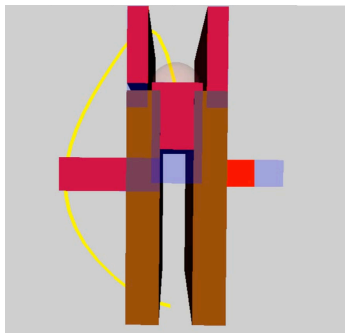

Fig. 20: Example run, guaranteed-safe policy (quadrotor maze). Initial/actually executed plan (yellow).

Visualizing an example run: For one sampled possible environment (displayed in red in Figures 19-21), we visualize the trajectories executed by our policy, a policy which seeks to plan guaranteed-safe trajectories [1], and an optimistic policy [27]. Our policy moves to the right and seeks to cut through the possibly unsafe region in the top right (Figure 19); when observing that the region is blocked, our policy switches to the blue contingency trajectory. On the other hand, the guaranteed-safe policy (Figure 20) seeks to avoid all possible constraints; as a result, while this policy never needs to switch to a contingency plan, it also ends up deterministically executing a higher-cost
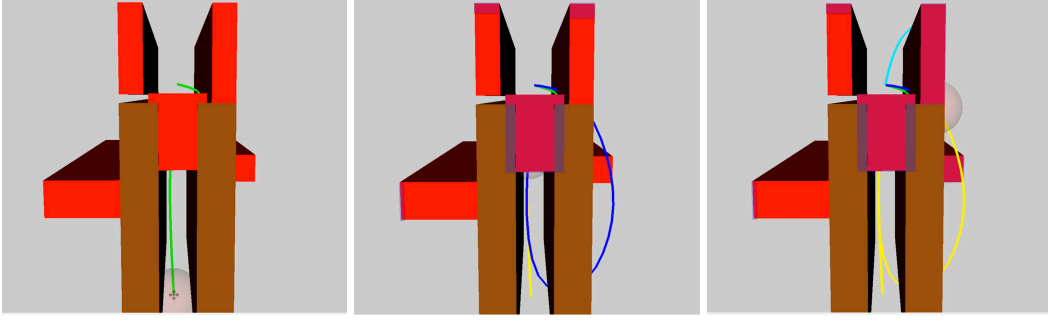
Fig. 21: Example run, optimistic policy (quadrotor maze). Initial plan (green), contingency plan 1 (blue), contingency plan 2 (cyan), actually executed plan (yellow).

trajectory. Finally, the optimistic policy in [27] explores the dead end between the brown obstacles and is forced to backtrack, yielding a higher cost compared to our policy.

F.6   Computation times

One challenge that our method faces when applied to real-time replanning is the computational intensity of online belief updates and online replanning of open-loop trajectories. First, we emphasize that if the assumptions in Section 5.4 hold, we can precompute the possible belief updates and contingency plans to avoid computing them online. If the assumptions are not satisfied, we will need to perform the computation online.

For belief updates, the computation time and number of measurements that can be updated depends heavily on the measurement type. For instance, updating the belief on the quadrotor maze example for a LiDAR scan with 30000 discretized points takes 1.4 seconds; LiDAR-type measurements are fast as each point is known safe or unsafe. However, contact measurements (as seen in the 7-DOF arm examples) are expensive as an unknown combination of the discretized points can be unsafe; modeling this requires the addition of many binary decision variables; it takes 30 minutes to incorporate a contact measurement with 300 discretized points. In this case, a further investigation of the tradeoff between accuracy and computation time based on the number of sampled possible contact measurements may lead to further computational gains.

The integer optimization variables are the key reason for slow $\mathcal{F}_\theta$ extraction in Algorithm 1. Thus, we are optimistic that we can speed up computation with parallelization (see Appendix C.3) and recent advances in fast mixed integer programming [40], which enjoy orders of magnitude speedup by learning efficient branching heuristics.

For open-loop planning, we note that the aforementioned fast mixed integer programming methods, as well as other work in warm-starting mixed integer programs, can be useful in reducing planning times for solving Problem 6 and other variants, as all of these variants are mixed integer programs, and the previous open-loop plan can serve as a good initalization for replanning. We also emphasize that we can reduce BTP planning times for the 7-DOF arm examples to around 15 seconds (as in the original BTP paper [36]) by precomputing arm swept volumes along roadmap edges and by employing lazy collision checking.