Learning Parametric Constraints in High Dimensions from Demonstrations

Glen Chou, Dmitry Berenson, Necmiye Ozay
Department of Electrical Engineering and Computer Science
University of Michigan, Ann Arbor, Michigan, 48910
Email: {gchou, dmitryb, necmiye}@umich.edu

I. Introduction

Inverse optimal control and inverse reinforcement learning (IOC/IRL) [6] can enable robots to perform complex goal-directed tasks by learning a cost function which replicates the behavior of an expert demonstrator when optimized. However, planning for many robotics and automation tasks also requires knowing constraints, which define what states or trajectories are safe. Existing methods learn local trajectory-based constraints [4, 5] or a cost penalty to approximate a constraint [2], neither of which extracts states that are guaranteed unsafe for all trajectories. In contrast, recent work [3] recovers a binary representation of globally-valid constraints from expert demonstrations by sampling lower cost (and hence constraint-violating) trajectories and then recovering a constraint consistent with the data by solving an integer program over a gridded constraint space. The learned constraint can be then used to inform a planner to generate safe trajectories connecting novel start and goal states. However, the gridding restricts the scalability of this method to higher dimensional constraints. The contributions of this workshop paper are twofold:

- By assuming a known parameterization of the constraint, we extend [3] to higher dimensions by writing a mixed integer program over parameters which recovers a constraint consistent with the data.
- We evaluate the method by learning a 6-dimensional pose constraint on a 7 degree-of-freedom (DOF) robot arm.

II. Preliminaries and Problem Statement

We consider a state-control demonstration ($\xi_x^* \doteq \{x_0, \ldots, x_T\}, \xi_u^* \doteq \{u_0, \ldots, u_{T-1}\}$) which steers a control-constrained system $x_{t+1} = f(x_t, u_t, t), u_t \in \mathcal{U}$ for all t, from a start state x_0 to a goal state x_T, while minimizing cost $c(\xi_x, \xi_u)$ and obeying safety constraints $\phi(\xi) \doteq \phi(\xi_x, \xi_u) \in \mathcal{S}$ and $\bar{\phi}(\xi) \doteq \bar{\phi}(\xi_x, \xi_u) \in \bar{\mathcal{S}}$. Formally, a demonstration solves the following problem[1]:

Problem 1 (Demonstrator's problem)**.**

$$
\begin{array}{ll}
\underset{\xi_x, \xi_u}{\text{min}} & c(\xi_x, \xi_u) \\
\text{s.t.} & \phi(\xi_x, \xi_u) \in \mathcal{S} \subseteq \mathcal{C} \\
& \bar{\phi}(\xi_x, \xi_u) \in \bar{\mathcal{S}} \subseteq \bar{\mathcal{C}}
\end{array}
$$

Here, $\phi(\cdot)$ and $\bar{\phi}(\cdot)$ are known functions mapping (ξ_x, ξ_u) to some constraint spaces $\mathcal{C}$ and $\bar{\mathcal{C}}$, where subsets $\mathcal{S} \subseteq \mathcal{C}$ and $\bar{\mathcal{S}} \subseteq \bar{\mathcal{C}}$ are considered safe. In particular, $\bar{\mathcal{S}}$ is known and represents the set of all constraints known to the learner.

In this paper, we consider the problem of learning the unsafe set $\mathcal{A} \doteq \mathcal{S}^c$, given N_s demonstrations $\{(\xi_x^*, \xi_u^*)_i\}_{i=1}^{N_s}$, each with different start and goal states. We assume that the dynamics, control constraints, and start and goal constraints are known and are embedded in $\bar{\phi}(\xi_x, \xi_u) \in \bar{\mathcal{S}}$. We also assume the cost function $c(\cdot, \cdot)$ is known.

[1] Details for continuous-time and suboptimal demonstrations are in [3].

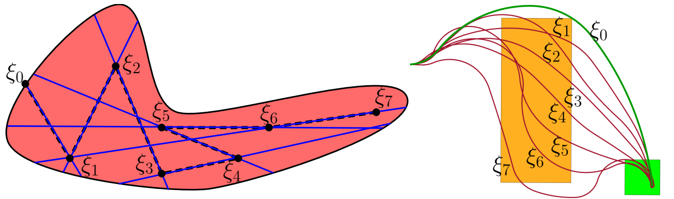

Fig. 1.**Left**: hit-and-run in trajectory space. **Right**: associated trajectories (red: lower-cost, thus violating the orange constraint; green: demonstration) in $\mathcal{C}$.

III. Learning Constraints on a Grid

We review [3], which reduces the ill-posedness of the constraint learning problem by using the insight that each safe, optimal demonstration induces a set of lower-cost trajectories that must be unsafe. These unsafe trajectories are sampled (Section III-A) and used with the demonstrations to reduce the number of consistent unsafe sets. Then, an integer program is used to find a gridded representation of $\mathcal{A}$ consistent with both safe and unsafe trajectories (Section III-B).

A. Sampling lower-cost trajectories

We are interested in sampling from the set of lower-cost trajectories which are dynamically feasible, satisfy the control constraints, and have fixed start and goal state x_0, x_T:

$$
\begin{aligned}
\mathcal{A}_\xi \doteq \{(\xi_x, \xi_u) \mid c(\xi_x, \xi_u) < c(\xi_x^*, \xi_u^*),\ x_{t+1} = f(x_t, u_t),\ \forall t, \\
u_t \in \mathcal{U},\ \forall t,\ \xi_x(0) = x_0,\ \xi_x(T) = x_T\}
\end{aligned}
\tag{1}
$$

Each trajectory $\xi_{\neg s} \in \mathcal{A}_\xi$ is unsafe, since the optimal demonstrator would have provided any safe lower-cost trajectory, and thus at least one state in $\xi_{\neg s}$ belongs to $\mathcal{A}$. We sample from $\mathcal{A}_\xi$ using hit-and-run [1, 3] (see Figure 1), providing a uniform distribution of samples in the limit. Furthermore, if the demonstrator is boundedly suboptimal and satisfies $c(\xi_x^{\text{dem}}, \xi_u^{\text{dem}}) \leq (1 + \delta)c(\xi_x^*, \xi_u^*)$ for known δ, guaranteed unsafe trajectories can be sampled by replacing $c(\xi_x, \xi_u) < c(\xi_x^*, \xi_u^*)$ in (1) with $c(\xi_x, \xi_u) < \frac{c(\xi_x^{\text{dem}}, \xi_u^{\text{dem}})}{1+\delta}$ [3].

B. Recovering the gridded unsafe set

As the constraint is not assumed to have any parametric structure, the constraint space $\mathcal{C}$ is gridded into G cells $z_1, \ldots, z_G$, and we recover a safety value for each grid cell $\mathcal{O}(z_i) \in \{0, 1\}$ which is consistent with the N_s safe and $N_{\neg s}$ sampled unsafe trajectories by solving the integer problem:

Problem 2 (Grid-based constraint recovery problem)**.**

$$
\begin{array}{lll}
\text{find} & \mathcal{O}(z_1), \ldots, \mathcal{O}(z_G) \in \{0,1\}^G & \\
\text{s.t.} & \displaystyle\sum_{z_i \in \{\phi(\xi_{s_j}^*(1)), \ldots, \phi(\xi_{s_j}^*(T_j))\}} \mathcal{O}(z_i) = 0, & \forall j = 1, \ldots, N_s \\
& \displaystyle\sum_{z_i \in \{\phi(\xi_{\neg s_k}(1)), \ldots, \phi(\xi_{\neg s_k}(T_k))\}} \mathcal{O}(z_i) \geq 1, & \forall k = 1, \ldots, N_{\neg s}
\end{array}
$$

Here, $\mathcal{O}(z_i) = 1$ if cell z_i is considered unsafe, and 0 otherwise. The first constraint restricts all cells that a demonstration passes through to be marked safe, while the second constraint restricts that for each unsafe trajectory, at least one grid cell it passes through is unsafe. Furthermore, denote as $\mathcal{G}_{\neg s}^z$ the set of guaranteed learned unsafe cells. One can check if cell $z_i \in \mathcal{G}_{\neg s}^z$ by checking the feasibility of Problem 2 with an additional constraint that $\mathcal{O}(z_i) = 0$ (forcing z_i to be safe).

IV. Learning Parametric Constraints

Suppose that the unsafe set can be described by some parameterization $\mathcal{A}(\theta) \doteq \{k \in \mathcal{C} \mid g(k, \theta) \leq 0\}$, where constraint state k is some element of $\mathcal{C}$, $g(\cdot, \cdot)$ is known, and θ are parameters to be learned. Then, another feasibility problem analogous to Problem 2 can be written to find a feasible θ consistent with the data:

Problem 3 (Parametric constraint recovery problem)**.**

$$\begin{array}{ll} \text{find} & \theta \\ \text{s.t.} & g(k_i, \theta) > 0, \quad \forall k_i \in \phi(\xi_{s_j}), \quad \forall j = 1, \ldots, N_s \\ & \exists k_i \in \phi(\xi_{\neg s_k}), \quad g(k_i, \theta) \leq 0, \quad \forall k = 1, \ldots, N_{\neg s} \end{array}$$

Denote $\mathcal{G}_s$ and $\mathcal{G}_{\neg s}$ as the set of guaranteed learned safe and unsafe constraint states. One can check if a constraint state $k \in \mathcal{G}_{\neg s}$ or $k \in \mathcal{G}_s$ by enforcing $g(k, \theta) > 0$ or $g(k, \theta) \leq 0$, respectively, and checking feasibility of Problem 3. Crucially, $\mathcal{G}_{\neg s}$ and $\mathcal{G}_s$ are guaranteed underapproximations of $\mathcal{A}$ and $\mathcal{A}^c$ (for space, we omit the proof; c.f. [3]).

A particularly common parameterization of an unsafe set is as a polytope $\mathcal{A}(\theta) = \{k \mid H(\theta)k \leq h(\theta)\}$, where $H(\theta)$ and $h(\theta)$ are affine in θ. In this case, θ can be found by solving a mixed integer feasibility problem:

Problem 4 (Polytopic constraint recovery problem)**.**

$$\begin{array}{ll} \text{find} & \theta, \{b_s^i\}_{i=1}^{N_s}, \{b_{\neg s}^i\}_{i=1}^{N_{\neg s}} \\ \text{s.t.} & H(\theta)k_i > h(\theta) - M(1 - b_s^i), \quad b_{s_j}^i \in \{0,1\}^{N_h}, \\ & \displaystyle\sum_{i=1}^{N_h} b_{s_j}^i \geq 1, \forall k_i \in \phi(\xi_{s_j}), i = 1, \ldots, T_j, j = 1, \ldots, N_s \end{array} \tag{2a}$$

$$\begin{array}{l} H(\theta)k_i \leq h(\theta) + M(1 - b_{\neg s_k}^i)\mathbf{1}_{N_h}, \quad b_{\neg s_k}^i \in \{0,1\}, \\ \displaystyle\sum_{i=1}^{T_j} b_{\neg s_k}^i \geq 1, \quad \forall k_i \in \phi(\xi_{\neg s_k}), \quad \forall k = 1, \ldots, N_{\neg s} \end{array} \tag{2b}$$

where M is a large positive number and $\mathbf{1}_{N_h}$ is a column vector of ones of length N_h. Constraints (2a) and (2b) use big-M formulations to enforce that each safe constraint state lies outside $\mathcal{A}(\theta)$ and that at least one constraint state on each unsafe trajectory lies inside $\mathcal{A}(\theta)$.

A few remarks are in order:

- If the safe set is a polytope or if the safe set or unsafe set is a union of polytopes, a mixed integer feasibility program similar to Problem 4 can be solved to find θ. A more general case where $g(k, \theta)$ is described by a Boolean conjunction of convex inequalities can be solved using satisfiability modulo convex optimization [7].
- In addition to recovering sets of guaranteed learned unsafe and safe constraint states, a probability distribution over possibly unsafe constraint states can be estimated by sampling unsafe sets from the feasible set of Problem 3.
- For suboptimal demonstrations or imperfect lower-cost trajectory sampling, Problem 4 can become infeasible. To address this, slack variables can be introduced: replace constraint $\sum_{i=1}^{T_j} b_{\neg s}^i \geq s_k, s_k \in \{0,1\}$ and change the feasibility problem to minimization of $\sum_{k=1}^{N_{\neg s}}(1 - s_k)$.

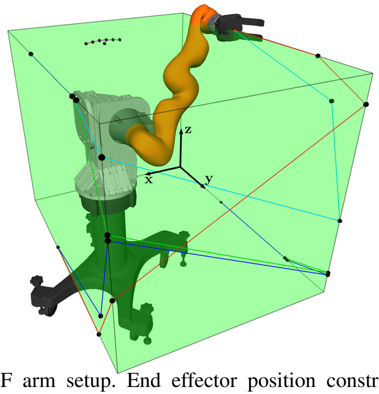

Fig. 2. 7-DOF arm setup. End effector position constraint (green box). Demonstrations (position component) color-coded to match with Figure 3.

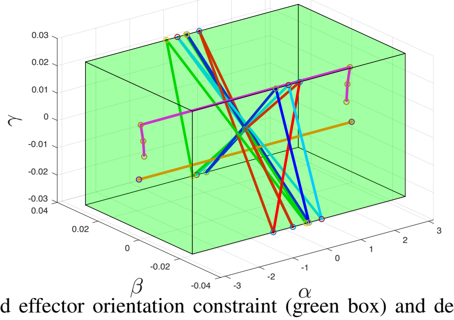

Fig. 3. End effector orientation constraint (green box) and demonstrations (orientation component).

V. Evaluation on 6D Constraint

In this example, we learn a 6D hyper-rectangular pose constraint for the end effector of a 7-DOF Kuka iiwa arm. In this scenario, the robot's task is to pick up a cup and bring it to a human, all while ensuring the cup's contents do not spill and proxemics constraints are satisfied (i.e. the end effector never gets too close to the human). To this end, the end effector orientation (parametrized in Euler angles) is constrained to satisfy $(\alpha, \beta, \gamma) \in [\underline{\alpha}, \bar{\alpha}] \times [\underline{\beta}, \bar{\beta}] \times [\underline{\gamma}, \bar{\gamma}] = [-\pi, \pi] \times [-\frac{\pi}{60}, \frac{\pi}{60}] \times [-\frac{\pi}{60}, \frac{\pi}{60}]$ (see Figure 3), while the end effector position is constrained to lie in $(x, y, z) \in [\underline{x}, \bar{x}] \times [\underline{y}, \bar{y}] \times [\underline{z}, \bar{z}] = [-0.51, 0.51] \times [-0.3, 1.1] \times [-0.51, 0.51]$ (see Figure 2). Six demonstrations optimizing $c(\xi_x, \xi_u) = \sum_{i=1}^{T-1} \|x_{i+1} - x_i\|_2^2$ are generated by solving trajectory optimization problems for the kinematic, discrete-time model in 7D joint space, where for each demonstration $T = 6$ and control constraints $u_t \in [-2, 2]^7$, for all t (see Figures 2, 3).

The constraint is recovered with Problem 4, where $H(\theta) = [I, -I]^\top$ and $h(\theta) = \theta = [\bar{x}, \bar{y}, \bar{z}, \bar{\alpha}, \bar{\beta}, \bar{\gamma}, \underline{x}, \underline{y}, \underline{z}, \underline{\alpha}, \underline{\beta}, \underline{\gamma}]^\top$. From this data, Problem 4 is solved in 1.19 seconds on a 2017 Macbook Pro and returns the true θ and $\mathcal{G}_s = \mathcal{S}$. $\mathcal{G}_s$ is efficiently recovered using the insight that the axis-aligned bounding box of any two constraint states in $\mathcal{G}_s$ must be contained in $\mathcal{G}_s$, since $\mathcal{G}_s$ is the union of axis-aligned boxes and therefore must also be an axis-aligned box.

VI. Conclusion

In this paper, we extend [3] to learn higher dimensional constraints by leveraging a known parameterization. We show that the constraint recovery problem for the parameterized case can be solved with mixed integer programming, and evaluate the method on learning a 6D pose constraint for a 7-DOF robot arm. Future work involves using learned constraints for probabilistically safe planning and developing safe exploration strategies and active demonstration-querying strategies to reduce the uncertainty in the learned constraint.

Acknowledgements

This work was supported in part by ONR grants N00014-18-1-2501 and N00014-17-1-2050, and NSF grants ECCS-1553873 and IIS-1750489.

References

[1] Yasin Abbasi-Yadkori, Peter L. Bartlett, Victor Gabillon, and Alan Malek. Hit-and-run for sampling and planning in non-convex spaces. In *AISTATS 2017*, 2017.

[2] Kareem Amin, Nan Jiang, and Satinder P. Singh. Repeated inverse reinforcement learning. In *NIPS*, pages 1813–1822, 2017.

[3] Glen Chou, Dmitry Berenson, and Necmiye Ozay. Learning constraints from demonstrations. *Workshop on the Algorithmic Foundations of Robotics (WAFR)*, abs/1812.07084, 2018. URL http://arxiv.org/abs/1812.07084.

[4] Changshuo Li and Dmitry Berenson. Learning object orientation constraints and guiding constraints for narrow passages from one demonstration. In *ISER*. Springer, 2016.

[5] N. Mehr, R. Horowitz, and A. D. Dragan. Inferring and assisting with constraints in shared autonomy. In *(CDC)*, pages 6689–6696, Dec 2016. doi: 10.1109/CDC.2016.7799299.

[6] Andrew Y. Ng and Stuart J. Russell. Algorithms for inverse reinforcement learning. In *ICML '00*, pages 663–670, San Francisco, CA, USA, 2000.

[7] Yasser Shoukry, Pierluigi Nuzzo, Alberto L. Sangiovanni-Vincentelli, Sanjit A. Seshia, George J. Pappas, and Paulo Tabuada. SMC: satisfiability modulo convex programming. *Proceedings of the IEEE*, 106(9):1655–1679, 2018.