Human Demonstrations Enable Efficient Solutions to Sequential Manifold Planning Problems

Breanne Crockett*
University of Colorado Boulder
breanne.crockett@colorado.edu

Carl L. Mueller*
University of Colorado Boulder
carl.mueller@colorado.edu

Bradley Hayes
University of Colorado Boulder
bradley.hayes@colorado.edu

***Abstract*—Constrained motion planning algorithms generate solutions to planning problems that require robots to adhere to rigid behavioral restrictions. Traditionally, these problems are segmented and framed as requiring adherence to a single set of constraints throughout, a limiting compromise that results in motion planning over a single manifold. When a trajectory must comply with multiple sets of constraints in a single motion planning problem, each constraint set defines an implicit manifold within the planning space and solutions must sequentially traverse manifold intersections. This is known as a Sequential Manifold Planning Problem (SMPP). The choice of manifold intersection points plays a critical role in solving SMPPs, as a particular intersection point may not admit a path to a subsequent constraint manifold, preventing motion planners from finding solutions in reasonable or even finite time. Many works assume *intersection point independence*, requiring all intersection points to lead to viable solutions. We show how Learning from Demonstration models intrinsically define an SMPP and contribute an algorithm for *Intersection Point Dependence Relaxation* using distributions extracted from these models near constraint set transitions. These distributions, learned from human demonstrations, supply candidate points for an optimization process to identify intersection points that admit solutions, solving SMPPs with greater efficiency than uninformed approaches and relaxing intersection point dependence even when the demonstrator is noisy (i.e., out of adherence to constraints).**

***Index Terms*—Learning from Demonstration; Constrained Motion Planning**

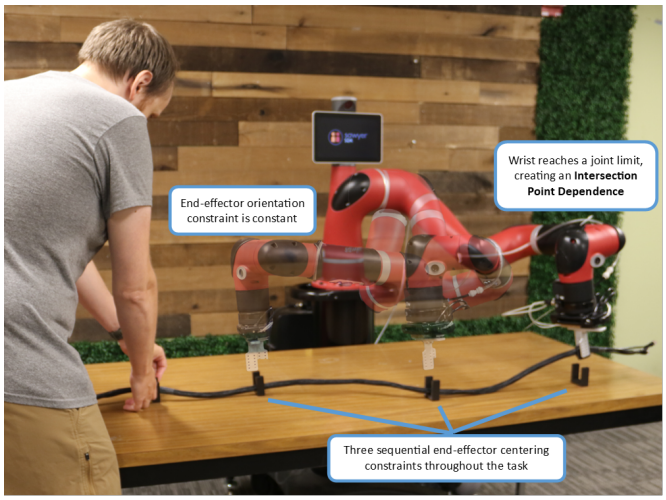


Fig. 1: A robot manipulator executing a cable-routing task in a shared workspace. The task has changing constraints that define a Sequential Manifold Planning Problem. The end-effector orientation is a constrained throughout the task to avoid twisting the cable. Three centering constraints require the manipulator to sequentially visit each cable guide.

I. Introduction

Sampling-based motion planning algorithms are widely used in robotics to generate trajectories that accommodate a variety of objectives in their search (e.g. collision avoidance) to produce desirable behavior. However, special accommodation is required to support adherence to hard constraints (e.g., planning on a surface). In human-robot interaction scenarios, constraints are critical for ensuring safe and effective collaboration between humans and robots, where the consequences of constraint violations may include safety risks or task failures. Constrained motion planning techniques exist to address this problem, but they traditionally focus on a single set of constraints that forms a single (implicitly defined) manifold during planning. In many HRI tasks, the set of constraints that must be satisfied can change dynamically as the interaction unfolds—for example, as a robot transitions between subtasks or adapts to human actions. Adherence to multiple sets of constraints that change within a single trajectory is more challenging, as the planner must find both the intersections between implicit manifolds and a sequence that eventually leads to the goal state of the planning problem. This is called a Sequential Manifold Planning Problem (SMPP) [15], and they are quite common in robotics domains (Figure 1). While existing motion planning work in this area has focused on the *intersection independent* class of SMPP, where all manifold intersection points lead to viable solutions, we introduce a method applicable to all SMPPs.

In this work, we show how robot Learning from Demonstration (LfD) techniques that incorporate constraints define an SMPP. Despite the utility of demonstrations, many types of constraints require precise angles, forces, or positions over time that humans are unlikely to be able to adhere to. We show how the resultant learned models of even noisy or imprecise human demonstration can be used to obtain a set of distributions that allow for efficiently solving SMPPs, guiding the selection of constraint manifold intersection points and providing a method to bias sampling toward constraint-compliant candidate points for planning. Our contributions are as follows:

- A method for combining learned models of human demonstration data with constrained motion planning methods to both define and efficiently solve SMPPs without the intersection point independence assumption through **Intersection Point Dependence Relaxation**.
- A quantitative evaluation showing how learned distributions increase sampling efficiency for constrained motion planners that use Jacobian projection-based methods for producing on-manifold samples. This supports the above contribution by increasing planning efficiency in scenarios that use human demonstration data that need only be *close* to constraint-compliant.

II. Preliminaries

A. Constrained Learning from Demonstration

Robot Learning from Demonstration (LfD) methods learn models from observation-borne data to create controllers that imitate the underlying skill being demonstrated [4]. Humans provide these data through some mode of demonstration (i.e., kinesthetic, teleoperated, or observed) [2]. Our work leverages information already captured by constrained LfD approaches [3, 38, 1]. These data are used to learn a sequential waypoint (i.e. keyframe) model of a skill through the clustering of demonstrated trajectories. Keyframe models produce a sequence of waypoints that serve as a coarse trajectory for the robot to follow, with inter-waypoint motion specified by a feedback controller, motion planner, or trajectory optimization technique. For constrained motion planning, we augment the traditional keyframe model [1] by enabling demonstrators to annotate constraints over keyframe segments during a demonstration, as in [38]. These constraints (e.g., "the teapot must remain upright until over the cup") provide the learner with additional information that helps capture a more complete model of the task.

Keyframe LfD algorithms produce a graph-based skill representation useful for the execution of (constrained) robot behaviors. Each node (keyframe) represents a distribution learned from demonstration data, with directed edges derived from temporal sequence. Throughout this work we refer to keyframes that capture a change in constraints as *transition* keyframes and keyframes that capture the continuation of constraints between transitions as *intermediate* keyframes [38].

B. Constrained Motion Planning

Sampling-based motion planning algorithms, such as probabilistic roadmaps [22], rapidly-exploring random trees [35], and their variants provide a robust toolkit to solve many planning problems [33, 11]. However, certain problems that place hard constraints on the motion of the robot challenge these planners by reducing the feasible solution space [27]. This reduced planning space is defined by Equation 1, which specifies a manifold (see Berenson et al. [7] and Stilman [43]). In Equation 1, satisfying the system $C(q) = 0$ for cost function C indicates a configuration point q is constraint-compliant. Notably, if this system requires a mapping of the point q into some other space (e.g. task space), the system then defines an *implicit* manifold space embedded within the configuration space Q [28]. Otherwise, the space M_C is *explicit*.

$$M_C = \{q \in Q : C(q) = 0\} \tag{1}$$

To enable sampling-based methods, the majority of constrained motion planning techniques rely on the ability to produce constraint-compliant configuration points during sampling, a process orthogonal to the planning algorithm [27, 29]. Unfortunately, it is not usually possible to sample points directly from implicit constraint manifolds [43, 7, 29].

To generate points on or near constraint manifolds, Jacobian-based gradient projection is often used, a sampling procedure that produces constraint-compliant points while providing guarantees of probabilistic completeness [7, 29, 43]. Jacobian-based gradient projection is an iterative process that first computes an error signal for each point. If the error signal is computed in task space, then it is mapped into configuration space. The error is used to push a configuration point towards constraint compliance within an ϵ-ball tolerance. We represent task space constraints using the Task Space Region framework of Berenson et al. [7] and use the associated Constrained Bidirectional RRT (CBiRRT2) motion planner that employs the Jacobian projection method with a mapping of task space error signals to configuration space.

C. Biased Sampling in Motion Planning

One of the strategies used by sampling-based motion planning algorithms to achieve probabilistic completeness is stochastic sampling, usually through uniform random sampling of the planning space [45, 32, 34, 23, 6, 7, 19, 33]. However, a drawback of uniform sampling is the wasted effort of sampling points that do not contribute to a solution. In the case of constrained motion planning, this adds additional unnecessary computation during the projection process. A common approach to overcome this issue is to bias sampling toward a distribution that more closely covers an expected solution space with a small fraction sampled uniformly to retain probabilistic completeness [19, 8, 49, 46, 24, 10, 9].

D. Sequential Manifold Planning

While constrained motion planning algorithms produce feasible trajectories over a single set of constraints, many real-world problems require planning over multiple changing constraint sets. Task and motion planning (TAMP) methods (also known as multimodal planning) treat these sequential problems as solving for connected sequences of subtasks, each differentiated by environmental conditions [16, 17]. The combined solutions to the sequence of subtasks create a solution to the global task or motion planning problem, just as in SMPPs.

TAMP methods must balance between planning at the task level and planning at the motion level where each plan informs the other. Dantam et al. introduce a method for iteratively refining a task and motion plan by alternating generation of constraint-compliant task plans and feasible motion plans [13, 14]. Similarly, Kaelbling and Lozano-Pérez introduce

a method that incorporates task and motion planning with planning for perceptual actions that reduce uncertainty [21]. With many sampling-based TAMP methods, the ability to sample efficiently has significant impact on the efficiency of the method. [48]. Consequently, Kingston et al. introduce a method that improves sample-efficiency by using experience to solve foliated multi-modal problems[31]. Our work improves sample efficiency similarly via learning from demonstrations.

Switching between modes in most multimodal planners occurs under specific conditions, such as contact events or precise task changepoints. In other words, transitions between modes are usually pre-defined points that are associated with a change in constraints, agent behavior, or environment. For example, in Englert et al. [15] intersection points are chosen as discrete contact events such as picking up an object. In contrast to prior work that requires explicit changepoint specification [30, 26], our proposed method learns these changepoints as a combination of implicit constraint manifold intersections, user demonstrations, and explicit keyframe constraint annotations.

E. Intersection Point Dependency

SMPPs are particularly difficult because a solution trajectory must traverse a sequence of intersecting constraint manifolds, each representing a unique set of constraints applied during a portion of the trajectory. To produce trajectories following these constraints, points on the intersections of adjacent overlapping manifolds must be chosen such that intersection points on subsequent manifolds are reachable until a goal state is achieved. These general SMPPs are challenging because planning for discrete changes in constraints may not rely on obvious changepoint events, for example if the constraint set changepoints are only defined through keyframes in an LfD planning graph. Providing a solution to this class of problem is a main contribution of this work: a method that uses the same annotated keyframe LfD models that implicitly define a generalized SMPP to assist with solution generation.

The constraint manifolds that specify an SMPP may be disjoint or foliated [15, 31, 37, 25]. A foliated manifold is composed of submanifolds or leaves, where each leaf does not intersect with any other leaves. A prior or subsequent manifold might only intersect with a subset of these leaves, defining *Intersection Point Dependence* (IPD). Choosing a leaf from each set of constraint manifolds that intersects with adjacent constraint manifolds is key to solving the complete SMPP [18]. Our method uses human demonstration data as a heuristic to relax IPD SMPPs into intersection-point independent (IPI) SMPPs. Adopting the style of figures of Englert et al. [15], Figure 2 provides an example SMPP for both intersection independent and intersection dependent cases.

III. IPD-Relaxation from Human Demonstration

We propose that **learned models from human demonstrations of approximate solutions to an SMPP provide enough of a heuristic to relax an intersection point dependent SMPP to an intersection point independent SMPP with high probability**, reducing the set of possible intersection

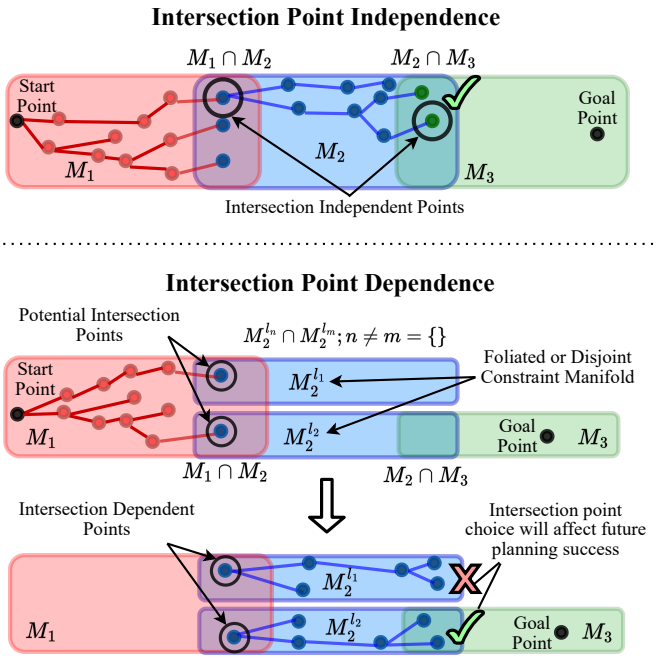


Fig. 2: Top: Planning success in Intersection Point Independent (IPI) SMPPs does not depend on the choice of intersection point. Bottom: An Intersection Point Dependent (IPD) SMPP where some intersection points result in planning failure.

points to viable choices only. We introduce an optimization process that directs the appropriate choice of intersection points using this information, in conjunction with biased sampling, such that the planner efficiently produces feasible solutions. We refer to this entire process as **Intersection Point Dependence Relaxation**.

A. ρ-usefulness and the Ω-set

To relax an IPD SMPP to an IPI SMPP, one must restrict the set of points drawn from manifold intersections strictly to those that can result in a feasible planning solution. We take inspiration from the definition of δ-usefulness in Rakita et al. [41] to define ***ρ*-useful points** as the set of points Q_ρ that are distance at most ρ from a solution path τ_i for a planning problem on manifold M_i given a planning space $C_{M_i} \cap C_{free,i}$ for the ith planning segment. Mathematically, ρ-useful points are $Q_\rho = \{q \in C_{M_i} \cap C_{\text{free}};\ d(q, \tau_i) \leq \rho\}$. Calculating an optimal distance for ρ is as difficult as solving the SMPP, but in practice setting it to a value that encompasses the distance between an approximately compliant demonstration and an exactly complaint trajectory is empirically straightforward and forgiving. We propose using a distribution D learned from demonstration data to bias the sampling such that $P(q \in Q_\rho; x \sim D) >> P(x \in Q_\rho; x \sim Q_{free})$: the probability that a point drawn from the learned distribution D is in Q_ρ is much greater than if the point were drawn uniformly from free configuration space Q_{free}.

The planner must select intersection points on a foliation leaf or disjoint constraint manifold intersection such that the next manifold intersection is reachable. We define an **Ω-**

point as a point on the intersection of sequentially adjacent manifolds that enables a feasible solution. The Ω-set is the collection of all Ω-points. Our insight is that demonstration data can either directly or nearly provide some of these points, informing their efficient selection during sampling. To generate Ω-points, we introduce **Omega Optimization**: a novel intersection point generation technique that uses multi-objective optimization to generate intersection points that maximize constraint compliance and minimize the distance from a corresponding constraint transition keyframe distribution. This process increases the likelihood that the generated intersection points are within the ρ-useful set and the Ω-set, conditions necessary for IPD-Relaxation.

B. IPD-Relaxation Formulation

Extending the SMPP formulation of Englert et al. [15], we define an IPD-Relaxed Sequential Manifold Planning Problem in Equation 2. C_* is the subset of the configuration space defined by some criteria $*$. $\boldsymbol{\tau}$ is a connected sequence of paths over manifolds $0...i...n$ that traverses through free configuration space. Constraint 1 ensures that a point S_i^{i+1} is an end point of one path on manifold i and the start point of the subsequent path on manifold $i+1$ (an *intersection point*). Constraint 2 dictates that this point is within the Ω_i^{i+1} set of $M_i \cap M_{i+1}$. Ω_i^{i+1} is the set of all configurations on the intersection of manifold M_i and subsequent manifold M_{i+1}, specifically within the foliation $M_{i+1}^{l^*}$ (with l^* unknown) that enables planning feasibility. The Ω_i^{i+1} set is inherently a local subset of the ρ-useful set as indicated by Constraint 3. Constraint 4 imposes that each segment be associated with its own collision-free space. Constraint 5 ensures the path is both collision-free and constraint complaint for manifold M_i.

Let $\boldsymbol{\tau} = (\tau_1, \ldots, \tau_n)$ for n planning segments *s.t.*

$$
\begin{aligned}
&1) \quad S_i^{i+1} = \tau_i(1) = \tau_{i+1}(0) && \forall_{i=1,\ldots,n-1} \\
&2) \quad S_i^{i+1} \in \Omega_i^{i+1} = \{q;\ C_{M_i} \cap C_{M_{i+1}^{l^*}}\} && \forall_{i=1,\ldots,n-1} \\
&3) \quad \Omega_i^{i+1} \subset \{q \in C_{M_i} \cap C_{\text{free},i};\ d(q,\tau_i) \leq \rho\} && \forall_{i=1,\ldots,n-1} \\
&4) \quad C_{\text{free},i+1} = \Upsilon(C_{\text{free},i}, S_i^{i+1}) && \forall_{i=1,\ldots,n-1} \\
&5) \quad \tau_i(s) \in C_{M_i} \cap C_{\text{free},i} && \forall_{i=1,\ldots,n} \\
& && \forall_{s\in[0,1]}
\end{aligned}
\tag{2}
$$

C. Leveraging Constrained-LfD Keyframe Distributions

Constrained-LfD model distributions can be characterized as one of two types [38]:

1) *Constraint Transition Keyframes*: Distributions at constraint set transitions, which we use to find productive constraint manifold intersections (Ω-points).
2) *Intermediate Keyframes*: Distributions learned between constraint transition keyframes, which we use to bias samples to adhere to the demonstrator's style.

We hypothesize the following effects of integrating LfD Keyframe distributions into SMPP solvers:

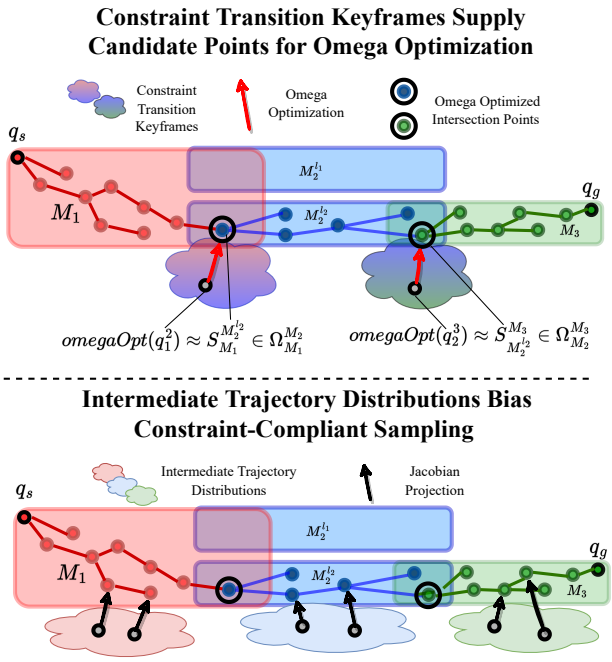


Fig. 3: A visual overview of this paper's contributions. *Top*: Constraint transition keyframes supply candidate points that are made constraint-compliant through Omega Optimization, facilitating IPD-Relaxation. *Bottom*: Intermediate trajectory distributions bias sampling during planning to improve planning time efficiency and adherence to demonstrator style.

1) Sample biasing using intermediate keyframes will both increase adherence to demonstrator style and boost planning efficiency by sampling points likelier to be from the ρ-useful set that are quicker to project onto constraint manifolds.
2) Sampling points from constraint transition keyframes will increase planner success by providing candidate points that optimize consistently into Ω-points using Omega Optimization, achieving IPD-Relaxation.

D. An Algorithm for IPD-Relaxation

Algorithm 1 uses a series of constraint-annotated keyframes (e.g., [38]) to achieve IPD-Relaxation. A constrained keyframe model K is passed as an input. It initializes a planning graph G_p to generate a sequence of connected plans that solve the IPD-Relaxed SMPP. The keyframe model K is traversed sequentially in reverse order using manifold intersections found by biasing point samples using the constraint transition keyframes (line 2-3), ignoring the intermediate keyframes.

A candidate intersection point s is drawn from distribution $k.D$ for a constraint transition keyframe $k \in K$ (line 5), which is passed to the Omega Optimization process (line 11), projecting the point onto the manifold intersection. When the optimization converges, if the resulting point o is valid and constraint-compliant it is added to the planning graph G_p alongside its associated constraints $k.C$ (line 14). Once an intersection point for each adjacent manifold pair are

Algorithm 1: IPD-Relaxation

Input: K ; // Planning Keyframe Sequence
Output: G_p ; // IPD Relaxed SMPP

1 $upcomingC$ = $K[-1].C$;
2 **for** k *in* $K.reversed()$ **do**
3 **if** isConstrained(k) **then**
4 **while** *! valid* **do**
5 $s \leftarrow$ sampleKeyframePoint($k.D$);
6 $combinedC$ = set($upcomingC$ + $k.C$);
7 **if** *constraintValid(s, combinedC)* **then**
8 $o \leftarrow s$
9 break;
10 **end**
11 $o, valid \leftarrow$ omegaOptimize($s, combinedC, k.D$);
12 **end**
13 $upcomingC = K.C$;
14 $G_p.addNode(o, k.C)$
15 **end**
16 **end**
17 **return** G_p

collected, a constrained motion planner (e.g., CBiRRT2 [7]) plans trajectory segments to create a global solution trajectory.

IV. Experiments

We evaluate our method in four settings, showing improved sampling efficiency and planning feasibility on IPD SMPPs when using human demonstrations as a learned heuristic. We do not evaluate a general audience's ability to provide demonstrations in this work: motion efficient demonstration data was sourced from robotics researchers with experience using each platform. Despite the demonstrators' familiarity with each robot platform, the provided demonstrations do not precisely adhere to constraints. Specifics on implementation parameters can be found in the appendix.

Independent Variables:

1) *Sampling*: Biased sampling from intermediate trajectory distributions vs. uniform sampling from configuration space for the CBiRRT2 planner. (Domains II, III, IV)
2) *Intersection Point Generation Mechanism*: Intersection points (i.e. the term 'o' in line 11, Algorithm 1) for use in planning will be generated by direct Keyframe sampling (KF), constraint-only optimization (CO) that projects sampled points to manifold intersections, and our method Omega Optimization (OO) that performs multi-objective optimization to balance distance to keyframes with projection to manifold intersections. (Domains II, III, IV)
3) *Collision Object*: Introduces a collision object that occludes the most common region demonstrated by users, creating a 'narrow corridor' condition. (Domain III only)

Metrics: Table I describes the metrics used for each domain. For evaluation Domain I, sampling time shows how biasing affects planning efficiency, as sampling points on constraint manifolds is the dominant source of computational overhead in constrained motion planning. For Domains II, III, and IV, we follow prior work [15] with metrics for Planning Success Rate, Path Length, and Planning Time. We introduce two metrics, *Adherence to Style (A2S)* and *Adherence to Function (A2F)*, to assess effectiveness in capturing aspects of the human-provided signal, as this contribution is a novel fusion of LfD with SMPPs. All trials were conducted on a PC with AMD Ryzen 9 5950X CPU with 32GB of RAM.

To provide a maximally fair basis for comparison, intersection points generated during IPD-Relaxation that are off-manifold and beyond the constraint-compliance tolerance (ϵ-ball) allowed by the CBiRRT2 planner are added into the RRT with a relaxed constraint tolerance to avoid simply failing to find a solution entirely. This is reflected in the Adherence to Function metric, as portions of the plan that traverse these off-manifold points will not adhere to the constraints.

A. Domain I - Constraint Demonstration for Biasing

In this evaluation domain, we use task-agnostic samples of constraint-compliant points sourced from three demonstrators on a Rethink Robotics Sawyer 7-DOF arm (Figure 6). The first constraint is an orientation constraint (e.g. holding a cup in an upright position). The second constraint is a line tracing constraint (e.g. adhesive application) restricting both orientation and position against a surface. The data from these demonstrations are used to fit KDE distributions. These distributions in turn produce candidate samples which are input into the projection operator in Section II-B.

B. Domain II - Planar Navigation on Explicit Manifolds

Five demonstrators each provided three teleoperated demonstrations on a holonomic 2D agent navigating from a start point to a goal point. The constraints are *explicit constraints* on the path and/or angle the agent must traverse. Two constraints restrict the XY position of the agent (blue and red in Figure 4). The third constraint (green in Figure 4) restricts the XY position as well as the angle of the agent as it must target the center of a black 'X' in the middle of the plane. The agent must adhere to the red, green, then blue constraints on its way to the goal. We provide a mathematical definition of constraints in the appendix.

We use a mixed-integer, non-linear, multiobjective program solved using the GEKKO solver [5] for Omega Optimization. While the constraints in this particular domain (including those with disjoint sets) can be explicitly solved for, choosing the optimal disjoint set depends on the set's proximity to a candidate point. The complete program is specified in the appendix.

C. Domain III - Manipulator Arm on Implicit Manifolds

Five demonstrators provided three kinesthetic demonstrations of a pouring task (Figure 5-left) on a Rethink Robotics Sawyer 7-DOF manipulator. For each demonstration, we record robot configuration and active constraints to generate

TABLE I: Description of Metrics for each Evaluation Domain

Metric	Abbreviation	Description	Domain
Sampling Time	n/a	Time (seconds) to sample 1000 constrained points	1
Success %	Success %	% successful planning trials	2, 3, 4
Path Length	PL	Euclidean path length (Domain 2: pixels, Domain 3: meters)	2, 3, 4
Adherence to Style	A2S	Dynamic Time Warping (DTW) distance in task space to 'gold' demonstration	2, 3, 4
Adherence to Function	A2F	% of planned trajectory constraint-compliant (no tolerance allowed)	2, 3, 4
Planning Time	PT	Time (seconds) needed to for successful plan	2, 3, 4

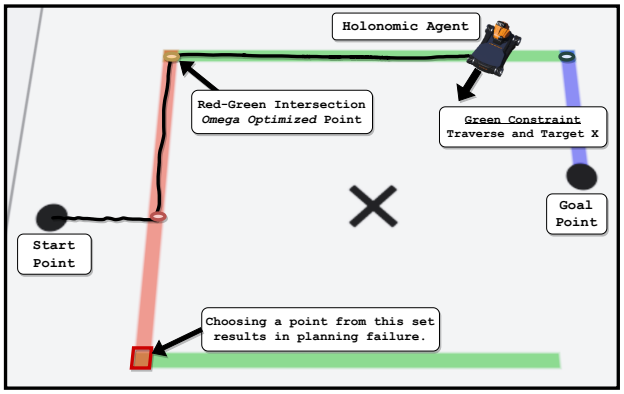


Fig. 4: Experiment environment for Domain II [36]. A holonomic agent (Kuka YouBot) moves from a start point to goal point traversing three sets of constraints to solve the SMPP.

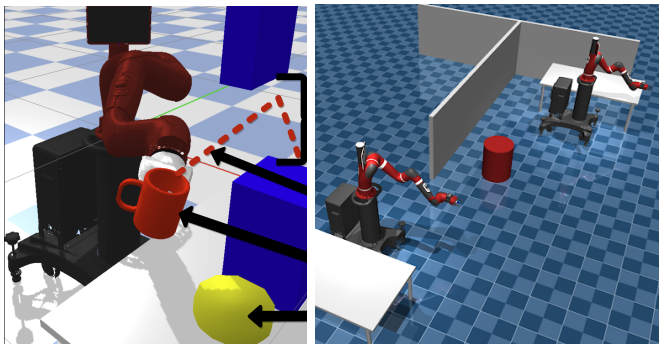


Fig. 5: Experiment environment for Domain III [12] and IV [47]. Left: Pouring scenario. The manipulator must observe an upright orientation until pouring, maintain a height above the lower collision object, and remain centered over the target once pouring. Right: Multi-robot object transfer scenario. The T-Shaped wall and mobile base handover position constraints induce IPD.

a CC-LfD keyframe model for each demonstrator. The task has three constraints: 1) an orientation constraint that requires the manipulator arm to hold a cup upright to avoid spills; 2) a height-restricting constraint to maintain a certain distance above the workbench; 3) a positional constraint requiring the end-effector to remain centered over a receptacle before pouring. Mathematical constraint definitions are defined in the appendix. While it is not possible to analytically verify whether this task is an SMPP with intersection point dependency without analytic manifold definitions, in 'collision object' trials the environment contains a fixed collision object creating a narrow gap near the receptacle that generally forces the end effector wrist to a joint limit, creating (with high probability) a disjoint manifold intersection between the first and third constraint. We use PANOC [42] for Omega Optimization, detailed in the appendix.

D. Domain IV - Multi-Robot Task on Implicit Manifolds

In this domain (Figure 5-right), a demonstrator provided three demonstrations via teleoperation. The robot state and active constraints (as specified by the demonstrator) were recorded at each time step. The task consists of two Rethink Robotics Sawyer manipulators and a 2-DOF holonomic mobile base. The robots must transport the cup from one table to the other by passing the cup from the first manipulator to the mobile base and then to the second manipulator. This task has three constraints: 1) An upright orientation constraint on the cup throughout the task, 2) a positional constraint on the first robot manipulator and the mobile base for the transfer of the cup from the manipulator to the mobile base, and 3) a positional constraint on the mobile base to be near the second robot manipulator for the transfer of the cup from the mobile base to the second manipulator. Domain IV uses the Omega Optimization function from Domain III for each manipulator.

V. Results and Discussion

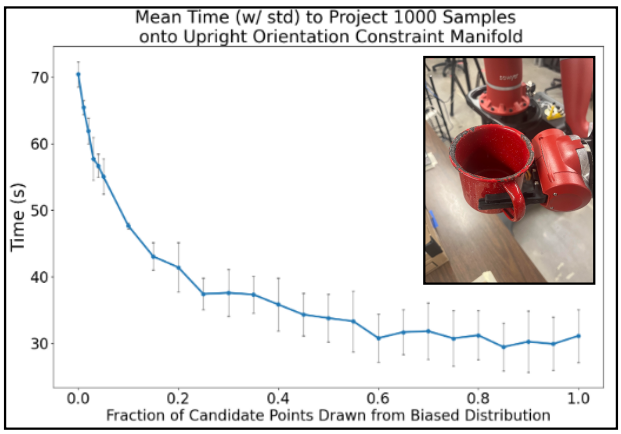
We conclusively confirm both hypotheses presented in Section III-C, demonstrating increased sample efficiency, success, and adherence to demonstrator style.

A. Domain I Results - Biased Sampling

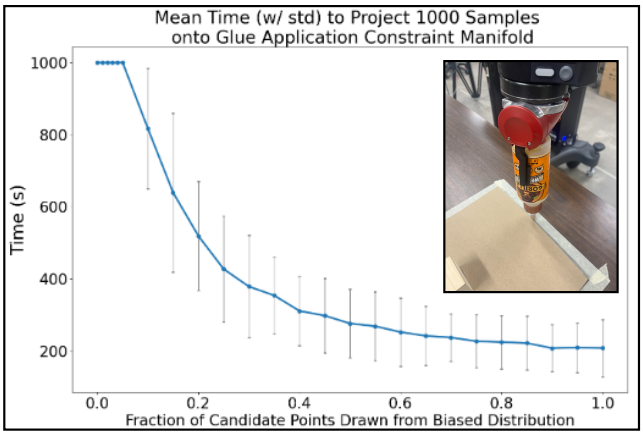
As seen in Figures 6a and 6b, distributions learned from user demonstrations that adhere (even if approximately) to constraints *substantially decrease* the sampling time needed to produce constraint-compliant points. This drop in sampling time exists when just 20% of the candidate points are drawn from a biased distribution. As the distributions produce candidate points that are close to the constraint manifold, less computation is needed to push the candidate point onto the manifold surface iteratively. Many constrained motion planning algorithms [7, 43, 39, 40, 20, 44] that use the Jacobian projection to generate constrained points could benefit from using human-provided demonstrations to bias sampling, with benefits that may transfer across dissimilar tasks or environments. Likewise, biased sampling is useful for planning problems requiring many constraint-compliant points for training [39, 40] or for roadmap-based techniques that need broad coverage of the planning space [29].

B. Domain II Results - Constrained Planar Navigation

Table II shows that the Omega Optimization (OO) intersection point generation mechanism performs best in most metrics given the explicit constraints and intersection point



(a) Domain I - Upright Orientation Constraint: Task-independent demonstrations of the cup in an upright orientation. Even a small amount of bias (20%) nearly halves required sampling and projection time.



(b) Domain I - Glue Application Constraint: Task-independent demonstrations of the robot agent holding the glue bottle in compliant orientations and height reduces sampling time. Sampling times are capped at 1000 seconds.

Fig. 6: Results for Domain I show the sampling time for biased sampling to find 1000 constraint-compliant points. Candidate samples are drawn either uniformly or from a biasing distributions to seed Jacobian projection in order to produce constraint-compliant points.

dependency. However, some metrics show little difference given the simplicity of the domain's environment.

Planning Success: Planning Success in Domain II relies on keyframe candidate points (KF & OO conditions). The CO conditions often choose the wrong intersection point, resulting in planning failure ($\leq$ 30% success for CO conditions).

Path Length: Across all conditions, Path Length values are highest (worst) in the CO condition as its compliant points do not necessarily follow the style of the demonstrated skill, often resulting in longer segments between intersection points. OO produced the best path length across all conditions.

Adherence to Style: CO results in poor A2S as intersection points are not coupled to demonstration data. A2S is the lowest (best) in the OO and KF conditions.

Adherence to Function: With biased sampling, the CO intersection point generation mechanism produced a slightly higher A2F in the rare cases that a successful plan was found. However, given the simplicity of this evaluation domain, the A2F values were all quite high ($\geq$ 91%). 100% A2F is unlikely as the interpolation between points can produce trajectory segments that are slightly off-manifold.

Planning Time: Planning times for the KF and OO conditions are nearly equivalent, with CO taking significantly longer. Using intermediate keyframe distributions to bias sampling decreases on-manifold planning time.

C. Domains III & IV Results

Tables III and IV depict the results for Domains III and IV, respectively. The Omega Optimization algorithm for intersection point generation performs best among the various conditions, achieving the best score across nearly every metric.

Planning Success: Planning success rates in both domains display the largest difference between intersection point generation conditions. In Domain III, Omega Optimization resulted in $\geq$ 88% planning success whereas all other intersection point generation methods never surpass 28%. The inclusion of a collision object that creates a narrow gap reduces the planning success across all conditions but reveals the resiliency of Omega Optimization in maintaining high planning success rates. Similarly, in the most difficult domain (Domain IV), Omega Optimization resulted in a substantially higher ($\geq$ 30%) planning success rate than other conditions ($\leq$ 2%).

Path Length: In Domain III, Path Length values are highest (poor) in the CO condition. Conditions using keyframe-derived candidate points significantly reduced path lengths in Domain III, whereas the Omega Optimization condition slightly reduced path length relative to the Keyframe Sampling conditions. The resultant path lengths are slightly shorter for the OO condition than the KF condition, which can produce slightly off-manifold-intersection points. This trend is less pronounced in Domain IV. In Domain IV, CO failed to produce a successful path in any of the 1030 trials, so path lengths could not be compared.

Adherence to Style: In Domain III, A2S for CO results in very high (poor) values as intersection points are not coupled to demonstration data. In this domain, the inclusion of a collision object results in the biggest contribution to A2S. The CO condition often produces paths that are constraint compliant but unusual compared to the style of the human demonstrators.

Adherence to Function: In Domain III, A2F is $\geq$99% across all conditions for both CO and OO intersection point generation methods within successful trials. However, the success rate for CO is substantially lower, with 27.2% as the best success percentage and the lowest 4.0% despite 99.89 and 99.94% A2F percentages respectively. No conditions can produce constraint-compliant points and segments perfectly, hence all conditions have $<$100% A2F.

Planning Time: Planning time is lower when using sampling bias. This suggests that intermediate trajectory distributions decrease planning time for successful planning events. In

TABLE II: Metrics across 50 trials per 5 demonstrators (250 trials) for Domain II. Bold: best within sampling-type group (gray/white)
**Only includes successful trials.

Conditions		Metrics				
Sampling	Intersection Point Generation	Success Rate (%)	Path Length (pixels)**	Adherence to Style (DTW Score)**	Adherence to Function (%)**	Planning Time (s)**
Biased	Keyframe Sampling	**100**	13758.11 ± 1602.04	21957.74 ± 5643.12	91.0 ± 0.01	**0.58 ± 1.81**
	Constraint Opt.	18.4	16223.32 ± 3928.009	50390.76 ± 34197.85	**95.0 ± 0.02**	2.05 ± 2.48
	Omega Optimization	**100**	**13400.16 ± 1629.37**	**18242.04 ± 3855.13**	93.3 ± 0.01	0.68 ± 0.09
Uniform	Keyframe Sampling	**100**	14387.65 ± 1481.11	21938.00 ± 5254.53	94.2 ± 0.942	**0.59 ± 1.40**
	Constraint Opt.	30	16682.55 ± 4107.92	82589.71 ± 51390.81	96.0 ± 0.01	5.67 ± 8.40
	Omega Optimization	**100**	**13693.93 ± 1472.88**	**19794.15 ± 4003.43**	**96.2 ± 0.00**	0.92 ± 1.84

TABLE III: Metrics across 25 trials per 5 demonstrators (125 trials) for Domain III. Bold: best within group (gray/white)
**Only includes successful trials.

Conditions			Metrics				
Collision Object	Sampling	Intersection Point Generation	Success Rate (%)	Path Length (m)**	Adherence to Style (DTW Score)**	Adherence to Function (%)**	Planning Time (s)**
Yes	Biased	Keyframe Sampling	2.4	12.51 ± 1.22	1488.36 ± 644.43	69.65 ± 14.01	**43.49 ± 3.18**
		Constraint Opt.	20.8	26.38 ± 5.87	8961.26 ± 3689.74	99.76 ± .58	49.93 ± 7.08
		Omega Optimization	**96.0**	**10.06 ± 2.26**	**1218.68 ± 395.34**	**99.99 ± .10**	44.18 ± 8.07
	Uniform	Keyframe Sampling	0.8	21.15 ± .00	2686.66 ± 0.00	82.28 ± 0.00	94.57 ± 0.00
		Constraint Opt.	4.0	63.10 ± 7.28	21921.38 ± 3935.41	99.77 ± .36	86.69 ± 11.87
		Omega Optimization	**88.0**	**12.23 ± 3.65**	**1474.80 ± 471.43**	**99.98 ± .12**	**50.50 ± 8.20**
No	Biased	Keyframe Sampling	11.2	11.57 ± 1.57	1285.38 ± 442.55	75.09 ± 12.90	**41.46 ± 1.28**
		Constraint Opt.	27.2	24.22 ± 4.54	7434.85 ± 1890.90	99.89 ± 0.38	48.51 ± 7.52
		Omega Optimization	**100**	**9.99 ± 2.09**	**1188.09 ± 362.50**	**99.94 ± .30**	42.79 ± 6.22
	Uniform	Keyframe Sampling	0.8	18.01 ± 4.65	2798.93 ± 1457.33	63.27 ± 14.08	75.75 ± 18.2
		Constraint Opt.	4.8	58.84 ± 11.86	18547.05 ± 8668.48	99.77 ± .34	92.30 ± 25.62
		Omega Optimization	**99.2**	**10.50 ± 2.62**	**1247.61 ± 433.55**	**99.98 ± .14**	**45.40 ± 8.48**

TABLE IV: Metrics across 1,030 trials for Domain IV. Bold: best within sampling-type group (gray/white)
**Only includes successful trials.

Conditions		Metrics				
Sampling	Intersection Point Generation	Success Rate (%)	Path Length (m)**	Adherence to Style (DTW Score)**	Adherence to Function (%)**	Planning Time (s)**
Biased	Keyframe Sampling	2.04	16.72 ± 1.17	**1799.50 ± 86.92**	**99.82 ± 0.48**	53.78 ± 4.99
	Constraint Opt.	0.0	-	-	-	-
	Omega Optimization	**33.40**	**16.70 ± 0.98**	1820.84 ± 82.32	99.94 ± 0.12	**51.95 ± 0.12**
Uniform	Keyframe Sampling	1.65	**16.23 ± 0.88**	**1817.41 ± 58.85**	99.90 ± 0.40	**52.75 ± 2.64**
	Constraint Opt.	0.0	-	-	-	-
	Omega Optimization	**31.85**	17.22 ± 1.98	1843.20 ± 145.87	**99.95 ± 0.11**	52.76 ± 2.77

Domain III, biased sampling has a normalizing effect for successful planning events across all other conditions. The trend is similar in Domain IV, albeit with smaller effect. This confirms that demonstration data is useful for more efficient planning, assuming approximately compliant demonstrations.

VI. Conclusion

These results demonstrate the effectiveness of leveraging human demonstrations as heuristics for constrained motion planning in Sequential Manifold Planning Problems, particularly those with intersection point dependence like many Task and Motion Planning domains. Our proposed method for intersection point dependence relaxation produces valid choices of intersection points far more often when seeded with points drawn from constraint transition keyframes built from human demonstrations, even when they are noisy or imperfect. This is directly evidenced in the illustrative Domain II and reinforced empirically by achieving top performance across all metrics in the more complex Domains (III and IV) in the Omega Optimization condition. Our work also shows that biasing sampling from LfD models is not only useful within the IPD-Relaxation algorithm, but also useful for other constrained motion planning methods and highlights a path for generalizing families of LfD models to inform efficient and feasible solutions to a wide variety of TAMP problems.

Limitations: Our proposed method surfaces several areas for further research. While the environments we selected test IPD-Relaxation, there could exist planning problems that shift the ρ-useful set away from the distributions of the CC-LfD models. This might negatively impact planning performance, as biased sampling would more often produce off-manifold points. Similarly, certain environmental conditions might force users to provide sub-optimal demonstrations, creating a similar effect of shifting points for Omega Optimization away from the Ω-set. Future work that could account for such shifts would make IPD-Relaxation generalize to noisier LfD models. Measuring the *helpfulness* of an LfD model for constraints or task families could improve the generality of our method.

References

[1] Baris Akgun, Maya Cakmak, Karl Jiang, and Andrea L Thomaz. Keyframe-based learning from demonstration. *International Journal of Social Robotics*, 4(4):343–355, 2012.

[2] Brenna D Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57 (5):469–483, 2009.

[3] Leopoldo Armesto, Vladimir Ivan, Joao Moura, Antonio Sala, and Sethu Vijayakumar. Learning constrained generalizable policies by demonstration. In *Robotics: Science and systems*, volume 13, 2017.

[4] Christopher G Atkeson and Stefan Schaal. Robot learning from demonstration. In *ICML*, volume 97, pages 12–20. Citeseer, 1997.

[5] Logan Beal, Daniel Hill, R Martin, and John Hedengren. Gekko optimization suite. *Processes*, 6(8):106, 2018. doi: 10.3390/pr6080106.

[6] Dmitry Berenson and Siddhartha S Srinivasaz. Probabilistically complete planning with end-effector pose constraints. In *2010 IEEE International Conference on Robotics and Automation*, pages 2724–2730. IEEE, 2010.

[7] Dmitry Berenson, Siddhartha Srinivasa, and James Kuffner. Task space regions: A framework for pose-constrained manipulation planning. *The International Journal of Robotics Research*, 30(12):1435–1460, 2011.

[8] Joshua Bialkowski, Michael Otte, and Emilio Frazzoli. Free-configuration biased sampling for motion planning. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1272–1279. IEEE, 2013.

[9] Valérie Boor, Mark H Overmars, and A Frank Van Der Stappen. The gaussian sampling strategy for probabilistic roadmap planners. In *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No. 99CH36288C)*, volume 2, pages 1018–1023. IEEE, 1999.

[10] Constantinos Chamzas, Anshumali Shrivastava, and Lydia E Kavraki. Using local experiences for global motion planning. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8606–8612. IEEE, 2019.

[11] Nikolaus Correll, Bradley Hayes, Christoffer Heckman, and Alessandro Roncone. *Introduction to Autonomous Robots: Mechanisms, Sensors, Actuators, and Algorithms*. MIT Press, 2022.

[12] Erwin Coumans. Bullet physics simulation. In *ACM SIGGRAPH 2015 Courses*, page 1. 2015.

[13] Neil T Dantam, Zachary K Kingston, Swarat Chaudhuri, and Lydia E Kavraki. Incremental task and motion planning: A constraint-based approach. In *Robotics: Science and systems*, volume 12, page 00052. Ann Arbor, MI, USA, 2016.

[14] Neil T Dantam, Swarat Chaudhuri, and Lydia E Kavraki. The task-motion kit: An open source, general-purpose task and motion-planning framework. *IEEE Robotics & Automation Magazine*, 25(3):61–70, 2018.

[15] Peter Englert, Isabel M Rayas Fernández, Ragesh K Ramachandran, and Gaurav S Sukhatme. Sampling-based motion planning on sequenced manifolds. *arXiv preprint arXiv:2006.02027*, 2020.

[16] Caelan Reed Garrett, Rohan Chitnis, Rachel Holladay, Beomjoon Kim, Tom Silver, Leslie Pack Kaelbling, and Tomás Lozano-Pérez. Integrated task and motion planning. *Annual review of control, robotics, and autonomous systems*, 4:265–293, 2021.

[17] Kris Hauser and Victor Ng-Thow-Hing. Randomized multi-modal motion planning for a humanoid robot manipulation task. *The International Journal of Robotics Research*, 30(6):678–698, 2011.

[18] Jiaming Hu, Shrutheesh R. Iyer, and Henrik I. Christensen. An experience-based tamp framework for foliated manifolds, 2023.

[19] Brian Ichter, James Harrison, and Marco Pavone. Learning sampling distributions for robot motion planning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7087–7094. IEEE, 2018.

[20] Léonard Jaillet and Josep M Porta. Path planning with loop closure constraints using an atlas-based rrt. In *Robotics Research*, pages 345–362. Springer, 2017.

[21] Leslie Pack Kaelbling and Tomás Lozano-Pérez. Integrated task and motion planning in belief space. *The International Journal of Robotics Research*, 32(9-10): 1194–1227, 2013.

[22] Lydia E Kavraki, Petr Svestka, J-C Latombe, and Mark H Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE transactions on Robotics and Automation*, 12(4):566–580, 1996.

[23] Lydia E Kavraki, Mihail N Kolountzakis, and J-C Latombe. Analysis of probabilistic roadmaps for path planning. *IEEE Transactions on Robotics and automation*, 14(1):166–171, 1998.

[24] Scott Kiesel, Ethan Burns, and Wheeler Ruml. Abstraction-guided sampling for motion planning. In *SoCS*, 2012.

[25] Jinkyu Kim, Inyoung Ko, and Frank C Park. Randomized path planning for tasks requiring the release and regrasp of objects. *Advanced Robotics*, 30(4):270–283, 2016.

[26] Zachary Kingston and Lydia E. Kavraki. Scaling multimodal planning: Using experience and informing discrete search. *IEEE Transactions on Robotics*, pages 1–19, 2022. doi: 10.1109/TRO.2022.3197080.

[27] Zachary Kingston, Mark Moll, and Lydia E Kavraki. Sampling-based methods for motion planning with constraints. *Annual review of control, robotics, and autonomous systems*, 1:159–185, 2018.

[28] Zachary Kingston, Mark Moll, and Lydia E Kavraki. Exploring implicit spaces for constrained sampling-based planning. *The International Journal of Robotics Research*, 38(10-11):1151–1178, 2019.

[29] Zachary Kingston, Mark Moll, and Lydia E Kavraki. Decoupling constraints from sampling-based planners. In

Robotics Research, pages 913–928. Springer, 2020.

[30] Zachary Kingston, Andrew M Wells, Mark Moll, and Lydia E Kavraki. Informing multi-modal planning with synergistic discrete leads. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3199–3205. IEEE, 2020.

[31] Zachary Kingston, Constantinos Chamzas, and Lydia E. Kavraki. Using experience to improve constrained planning on foliations for multi-modal problems. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, September 2021.

[32] Michal Kleinbort, Kiril Solovey, Zakary Littlefield, Kostas E Bekris, and Dan Halperin. Probabilistic completeness of rrt for geometric and kinodynamic planning with forward propagation. *IEEE Robotics and Automation Letters*, 4(2):x–xvi, 2018.

[33] S. M. LaValle. *Planning Algorithms*. Cambridge University Press, Cambridge, U.K., 2006. Available at http://planning.cs.uiuc.edu/.

[34] Steven M LaValle, James J Kuffner, BR Donald, et al. Rapidly-exploring random trees: Progress and prospects. *Algorithmic and computational robotics: new directions*, 5:293–308, 2001.

[35] Steven M LaValle et al. Rapidly-exploring random trees: A new tool for path planning. 1998.

[36] Olivier Michel. Cyberbotics ltd. webots™: professional mobile robot simulation. *International Journal of Advanced Robotic Systems*, 1(1):5, 2004.

[37] Joseph Mirabel and Florent Lamiraux. Manipulation planning: addressing the crossed foliation issue. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4032–4037. IEEE, 2017.

[38] Carl Mueller, Jeff Venicx, and Bradley Hayes. Robust robot learning from demonstration and skill repair using conceptual constraints. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6029–6036. IEEE, 2018.

[39] A. H. Qureshi, J. Dong, A. Choe, and M. C. Yip. Neural manipulation planning on constraint manifolds. *IEEE Robotics and Automation Letters*, 5(4):6089–6096, 2020.

[40] Ahmed H Qureshi, Jiangeng Dong, Asfiya Baig, and Michael C Yip. Constrained motion planning networks x. *IEEE Transactions on Robotics*, 2021.

[41] Daniel Rakita, Bilge Mutlu, and Michael Gleicher. Single-query path planning using sample-efficient probability informed trees. *IEEE Robotics and Automation Letters*, 6(3):4624–4631, 2021.

[42] Lorenzo Stella, Andreas Themelis, Pantelis Sopasakis, and Panagiotis Patrinos. A simple and efficient algorithm for nonlinear model predictive control. *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pages 1939–1944, 2017.

[43] Mike Stilman. Task constrained motion planning in robot joint space. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3074–3081. IEEE, 2007.

[44] Chansu Suh, Terry Taewoong Um, Beobkyoon Kim, Hakjong Noh, Munsang Kim, and Frank C Park. Tangent space rrt: A randomized planning algorithm on constraint manifolds. In *2011 IEEE International Conference on Robotics and Automation*, pages 4968–4973. IEEE, 2011.

[45] Petr Svestka. *On probabilistic completeness and expected complexity for probabilistic path planning*, volume 1996. Utrecht University: Information and Computing Sciences, 1996.

[46] Shawna Thomas, Marco Morales, Xinyu Tang, and Nancy M Amato. Biasing samplers to improve motion planning performance. In *Proceedings 2007 IEEE international conference on robotics and automation*, pages 1625–1630. IEEE, 2007.

[47] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 5026–5033. IEEE, 2012.

[48] William Vega-Brown and Nicholas Roy. Asymptotically optimal planning under piecewise-analytic constraints. In *Algorithmic Foundations of Robotics XII*, pages 528–543. Springer, 2020.

[49] Zhuping Wang, Yunsong Li, Hao Zhang, Chun Liu, and Qijun Chen. Sampling-based optimal motion planning with smart exploration and exploitation. *IEEE/ASME Transactions on Mechatronics*, 25(5):2376–2386, 2020.

I. APPENDIX

A. Implementation Details

All evaluations and implementations of the proposed IPD-Relaxation algorithm were executed on an AMD Ryzen 9 5950X 16-Core Processor with 32 GB of RAM and are implemented in Python except where noted below.

- Domain II uses a simplified constrained RRT planner with the following parameters: ϵ-tolerance: 50; extension distance: 10; max planning time; 60 seconds.
- Domains III and IV use the CBiRRT2 planner [1] with the following parameters: ϵ-tolerance: 0.15; q-step: 0.35; smoothing time: 5 seconds. In conditions that use biased sampling, 90% of points are sampled from the demonstration distributions and 10% of points are sampled uniformly from the configuration space. For global planning: Omega Optimization tolerance: 0.075; max SMPP segment planning time: 30 seconds. Omega Optimization tolerance is the allowed off-manifold error of the produced intersection point.
- In Domains II, III, and IV, a bandwidth of 0.15 for Kernel Density Estimation (KDE) was used to fit keyframes and intermediate trajectory data, following the CC-LFD algorithm from Mueller et al. [3]. These parameters balance performance against time efficiency and were consistent across all conditions.

In our experiments, we chose to analytically model constraints before capturing demonstrations as in [3]. Alternative implementations of the proposed framework could learn the constraints from the demonstrations [2]. Constraints do not necessarily need to be modelled analytically. Our framework only requires a distance function to the constraint for Omega Optimization.

B. Constraint Definitions for Experiment Domains II-IV

In this appendix, we specify on how constraints are defined for each domain in our experiments.

In Domain II, there are three sets of constraints on the robot, two sets with one position constraint and one set of a position constraint and an orientation constraint. In this domain, the constraints are explicitly defined in the robot's configurations space (x, y, θ) as follows. We label each set of constraints according to the color depicted in Figure 4. The black X in the center of the environment is located at the origin, while x_0, x_1, y_0, and y_1 denote the corners of the line tracing constraints. y_2 denotes the y-location of the goal.

$$
\begin{aligned}
\text{Red:}\quad & x = x_0 \\
& y_0 \leq y \leq y_1 \\
\text{Green:}\quad & x_0 \leq x \leq x_1 \\
& y = y_0 \text{ or } y = y_1 \\
& \theta = \begin{cases} \arctan\left(\frac{y}{x}\right) - \pi & \text{if } x > 0 \\ \arctan\left(\frac{y}{x}\right) & \text{if } x < 0 \\ -\frac{\pi}{2} & \text{if } x = 0 \text{ and } y > 0 \\ \frac{\pi}{2} & \text{if } x = 0 \text{ and } y < 0 \end{cases} \\
\text{Blue:}\quad & x = x_1 \\
& y_2 \leq y \leq y_1
\end{aligned}
\tag{1}
$$

In Domains III and IV, we model constraints using the Task Space Region framework introduced in [1]. Domain III has 3 constraints: 1) an orientation constraint to hold a liquid vessel in its upright position; 2) a height-restricting constraint where the end-effector must maintain a certain distance above the table; 3) a parameterized positional constraint requiring the end-effector to remain centered over a receptacle (Section IV-C). Domain IV has these constraints plus two positional constraints that describe the transfer point of the cup from one robot to the next (Section IV-D). We model these constraints in task space using XYZ for position and Roll-Pitch-Yaw for orientation. Each constraint is defined by a pose and the allowable bounds in each dimension. We specify the following positions in meters and orientation angles in degrees. For example, a positional constraint may consist of a position at $(1, 2, 4)$ and the bounds may be $[-1, 1]$ in x, $[-1, 1]$ in y, and $[0, 2]$ in z. This constraint dictates that $0 \leq x \leq 2$, $1 \leq y \leq 3$, and $4 \leq z \leq 6$. A particular dimension can be 'unbound' by selecting bounds that are larger than the planning environment. Similarly, an orientation constraint to keep a cup upright within an end-effector's grasp would be composed of an orientation $(0, 0, 0)$ and bounds for roll $(-5, 5)$, pitch $(-5, 5)$, and yaw $(-180, 180)$. In this example, an orientation of $(0, 0, 0)$ is the cup perfectly upright, and we allow the cup to tilt up to 5 degrees around the x- or y-axes while rotation around the z-axis is unconstrained.

C. Omega Optimization Program Details

Omega Optimization in Domain II: For Domain II, we use the following Omega Optimization program:

$$
\begin{aligned}
\min_{q=(q_x,q_y,q_\theta)} \quad & f(q) = w_1 * A * distIntersectionTop(q) \\
& \quad + w_2 * (1 - A) * distIntersectionBottom(q) \\
& \quad + w_3 * distToKeyframePoint(q, q_{kf}) \\
\textit{s.t.} \quad & \\
& 1) \quad withinLimits(q) \\
& 2) \quad A * distIntersectionTop(q) \leq \epsilon \\
& 3) \quad (1 - A) * distIntersectionBottom(q) \leq \epsilon \\
& 4) \quad A \in \{0, 1\} \\
& 5) \quad q_\theta = 360 - arctan(\frac{t_y - q_y}{t_x - q_x}) * 180/\pi
\end{aligned}
\tag{2}
$$

The function $distIntersectionTop(q)$ denotes the distance from the configuration q to the intersection of the red manifold

and the top of the two green manifolds depicted in Figure 4. Similarly, $distIntersectionBottom(q)$ denotes the distance from the configuration q to the intersection of the red manifold and the bottom of the two green manifolds. The binary integer component A (constraint 4) forces the optimizer to determine which of the two intersection choices minimizes the cost function (i.e., the optimizer knows about both intersections, but doesn't know which is correct a priori). The point q is initialized as q_{kf}, the candidate value that is generated according to the sampling technique of the experimental condition being run. For conditions in which keyframe points are not used, w_3 is set to zero to effectively remove the keyframe-associated term from the cost function. Constraints 2 and 3 enforce that the value chosen for the intersection point is within some ϵ tolerance. Constraint 5 ensures that the chosen value for the angle of the agent is the smallest value.

Omega Optimization in Domain III: As this domain has implicit manifold constraints, a more sophisticated optimization approach is used for Omega Optimization. Our approach is inspired by Rakita et al. [4], utilizing the PANOC optimization library for the Rust programming language [5]. In Equation 3, the cost function terms are each contained within a 'groove loss' function ('GL()') that combines linear and Gaussian terms with the effect of generating much lower costs near the optimal value for the wrapped function (see [4] for more details about this function and the standard parameters adopted by this work). This choice enables the integration of multiple terms into a multi-objective cost function and enables the use of finite-differencing methods for generating approximate gradients within the optimizer.

$$\begin{aligned} \min_{q} \quad f(q) = & \, w_1 * GL(DistTSRPosition(q)) \\ & + w_2 * GL(DistTSRQuat(q)) \\ & + w_3 * GL(DistKFPosition(q, q_{kf})) \\ & + w_4 * GL(DistKFQuat(q, q_{kf})) \end{aligned} \tag{3}$$

The $DistTSRPosition(q)$ and $DistTSRQuat(q)$ terms use the distance conventions of the Task Space Regions (TSRs) constraint framework introduced by [1]. This convention is used to easily define XYZ-position and Roll-Pitch-Yaw orientation constraints (i.e., task space constraints). We implement the distance for position and orientation separately to weigh the terms individually in the cost function. $DistKFPosition(q, q_{kf})$ and $DistKFQuat(q, q_{kf})$ restrain the optimizer so that the converged value does not stray too far from the candidate sample provided by the constraint transition keyframe distribution. These terms drive value in using keyframe distributions as a heuristic for generating Ω-set compliant points. The assumption is that finding an intersection point closer to the distribution is more likely to be in the Ω-set. For experiment conditions that perform optimization but do not use keyframe data, weights w_3 and w_4 are set to zero.

REFERENCES

[1] Dmitry Berenson, Siddhartha Srinivasa, and James Kuffner. Task space regions: A framework for pose-constrained manipulation planning. *The International Journal of Robotics Research*, 30(12):1435–1460, 2011.

[2] Glen Chou, Dmitry Berenson, and Necmiye Ozay. Learning constraints from demonstrations with grid and parametric representations. *The International Journal of Robotics Research*, 40(10-11):1255–1283, 2021. doi: 10.1177/02783649211035177. URL https://doi.org/10.1177/02783649211035177.

[3] Carl Mueller, Jeff Venicx, and Bradley Hayes. Robust robot learning from demonstration and skill repair using conceptual constraints. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6029–6036. IEEE, 2018.

[4] Daniel Rakita, Haochen Shi, Bilge Mutlu, and Michael Gleicher. Collisionik: A per-instant pose optimization method for generating robot motions with environment collision avoidance. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9995–10001. IEEE, 2021.

[5] Lorenzo Stella, Andreas Themelis, Pantelis Sopasakis, and Panagiotis Patrinos. A simple and efficient algorithm for nonlinear model predictive control. *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pages 1939–1944, 2017.