To appear in the Proc. of the IEEE Intl. Conf on Robotics & Automation, 2020.

Informing Multi-Modal Planning with Synergistic Discrete Leads

Zachary Kingston[1], Andrew M. Wells[1], Mark Moll[1], Julia Badger[2], and Lydia E. Kavraki[1]

***Abstract*— Robotic manipulation problems are inherently continuous, but typically have underlying discrete structure, e.g., whether or not an object is grasped. This means many problems are *multi-modal* and in particular have a continuous infinity of modes. For example, in a pick-and-place manipulation domain, every grasp and placement of an object is a *mode*. Usually manipulation problems require the robot to *transition* into different modes, e.g., going from a mode with an object placed to another mode with the object grasped. To successfully find a manipulation plan, a planner must find a sequence of valid single-mode motions as well as valid transitions between these modes. Many manipulation planners have been proposed to solve tasks with multi-modal structure. However, these methods require mode-specific planners and fail to scale to very cluttered environments or to tasks that require long sequences of transitions. This paper presents a general layered planning approach to multi-modal planning that uses a discrete "lead" to bias search towards useful mode transitions. The difficulty of achieving specific mode transitions is captured online and used to bias search towards more promising sequences of modes. We demonstrate our planner on complex scenes and show that significant performance improvements are tied to both our discrete "lead" and our continuous representation.**

I. Introduction

Many manipulation problems are *multi-modal* [1]. That is, the inherently continuous manipulation problem has some underlying discrete structure. Specifically, there is a finite set of high-level actions (e.g., picking up or placing an object) that each have a continuous infinity of instantiations (e.g., placing an object at some point on a table). Every specific instantiation is a *mode* that constrains the robot's motion differently (e.g., adding an end-effector constraint, or constraining where the robot can place a second object). *Transitioning* between modes is an essential component of manipulation planning—for example, arranging objects may require many grasps and placements. To solve manipulation problems automatically, a manipulation planner must reason over the sequence of modes the system must traverse. The planner needs to find feasible motions in each mode, as well as valid transitions from one mode to the next.

Many sampling-based multi-modal motion planners have been proposed [1]–[3], which can scale to high-dimensional problems. These are general solvers for manipulation planning problems, but they require specialized samplers or planners for each mode and cannot scale to problems that require many transitions. There are other manipulation planners [4]–[6], that are *layered* planning approaches; a "higher-level" planner finds what transitions to take, which then informs a "lower-level" planner that searches for feasible motions. These approaches are typically limited to specific problem domains such as pick-and-place problems.

This work proposes a method for highly general and efficient multi-modal planning, using insights from layered planning while avoiding unnecessary discretization. Inspired by SyCLoP [7], we propose a synergistic layered planner based on prior work in multi-modal motion planning [1]. This planner uses a novel weighting scheme in each iteration of planning: first, a discrete planner provides a candidate sequence of mode transitions (a "lead"). A continuous planner follows the "lead", estimating the difficulty of transitioning between modes. If a transition succeeds or fails, weights are updated to inform future discrete planning and bias search towards promising avenues of exploration.

In contrast to prior work, our approach does not require specialized samplers or planners. Instead we use a general formulation of modes as manifold constraints and leverage a general single-mode planning framework [8]. This formulation captures pick-and-place manipulation as well as a broader set of scenarios, e.g., handrail climbing. We demonstrate our planner on a variety of complex scenes and show that significant performance improvements are tied to both our discrete "lead" and our continuous representation.

II. Related Work

Manipulation planning is a core problem in robotics [9], [10] and has been studied for decades [11]–[14]. We focus on approaches that use sampling-based planning, a powerful probabilistically-complete paradigm for motion planning able to scale to high-dimensional problems [15]–[17].

Planning in a single mode requires planning under *constraints*; we focus on geometric planning problems with *manifold constraints*. There are many approaches for planning under manifold constraints (e.g., trajectory optimization-based [18], [19] or sampling-based [20]). A survey of sampling-based techniques is given in [21]. For this paper, we use the constrained planning framework presented in [8].

Frequently manipulation planners use *layered* planning: a combination of planners that inform each others' searches. Layered planning is a heuristic to speed-up search by solving the problem at different levels of abstraction [7], [22]. One abstraction is to introduce a *symbolic* or *discrete* representation, e.g., PDDL [23] or workspace discretizations [7]. Many planners (e.g., [23], [24]) discretize continuous problems.

[1]ZK, AMW, MM, and LEK are with the Department of Computer Science, Rice University, Houston, TX, USA {zak, awells, moll, kavraki} at rice.edu. ZK is supported by NASA Space Technology Research Fellowship 80NSSC17K0163. AMW is supported by NASA Space Technology Research Fellowship 80NSSC17K0164. This work is supported in part by NSF 1718478 and Rice University Funds.

[2]NASA-Johnson Space Center, Houston, TX 77058 julia.m.badger at nasa.gov

Common discretizations include choosing from a finite set of placement locations or grasps.

A common abstraction used in manipulation planning is a *mode graph* [11]—a discrete structure encoding possible transitions between modes. In task planning, mode graphs are implicitly defined by what actions are possible given the current symbolic state (e.g., [6]). For problems with finite modes, mode graphs have been successfully applied to high-dimensional problems [25]–[27] and dynamic environments [28]. However, manipulation problems have a continuous infinity of modes, e.g., an object can be placed at any valid location on the plane of the table. For problems with a continuous infinity of modes, *transition graphs* [1] capture what transitions between which "classes" of modes are possible. Our planner uses an abstraction similar to a transition graph to capture allowable transitions (see Sec. IV-B).

Some planners discretize continuously infinite modes (e.g, [4], [24]), but if the discretization is not fine enough a plan will not be found. With an offline discretization, [4] used a "fuzzy" PRM over the mode graph to determine what transitions to take, with edge weights proportional to the time spent solving the single-mode planning problem from one transition to another, biasing search towards "easier" planning problems. Manipulation RRT [26], [27] also weights transitions between modes online, but is limited to a finite set of modes. Our approach uses a similar idea for biasing search, but works for continuous infinities of modes (see Sec. IV-B). Similarly, online adaptation has been used to bias search, e.g., approximating collisions [29], [30]. Our method estimates online the feasibility of transitions between modes.

Another approach is online discretization via sampling. Manipulation PRM [5], [31], [32] and ASYMOV [6], [33], [34] build a PRM over the transition manifold. Other planners [1], [35] build a tree of transitions in the continuous mode space, similar to how sampling-based planners discretize a continuous space. In the case of navigation among movable obstacles, a probabilistically-complete tree-based planner has been proposed [35]. These works require specialized samplers or only work for finite sets of modes. As a more general approach, [1] proposes a probabilistically-complete tree-based planner for multi-modal problems, which our work draws inspiration from. "Utility tables" are used by [1] to capture valuable transitions in multi-modal search, but these are computed offline and require specialized domain knowledge. Our approach estimates the value of transitions online (see Sec. IV-C). DARRTH [2], [36], [37] is multi-modal planning approach that takes a layered planning approach, but uses a simplified continuous planning domain rather than a discrete layer. In a similar vein, [3] proposed an asymptotically optimal multi-modal planner. These approaches utilize specialized samplers for transitioning between modes.

Task and Motion Planning (TMP) algorithms generally have hierarchical layers of planning, such as in [23], [38]–[40]. Our approach differs twofold: symbolic representation of geometry and bias towards short task plans. TMP algorithms generally reason over "actions": actions cause a transition into a new mode. However, many TMP algorithms use discretization to create a finite representation amenable to a task planner (e.g., [23]). Even without a fixed discretization, TMP algorithms plan for geometric values represented symbolically, such as with sampling (e.g., [40]) or optimization (e.g., [41]–[43]). Tied to this symbolic representation, TMP algorithms usually bias search towards short symbolic task plans. However, there may not be a solution corresponding to a short symbolic plan, e.g., moving an object through a tight passage requiring many regrasps (see Fig. 5). Our approach deals with the geometry directly, avoiding this bias.

Our approach takes inspiration from SYCLOP, a general framework which uses a "lead" path through a discretization of the workspace [7]. In SYCLOP, planning is informed of promising avenues of exploration via discrete search, while discrete search is informed by feedback from motion planning. Our approach uses discrete leads through possible mode transitions to inform planning; successes and failures in planning inform the discrete layer. Notably, our framework and other manipulation planners based on SYCLOP can avoid the backtracking found in most TMP frameworks [44], but we do not address tasks with temporal goals, e.g., [45].

III. PRELIMINARIES

In this work, we consider manipulation planning problems that are *multi-modal*. These are composed of a finite set of *mode families*, each containing a continuous infinity of *modes* (as in [1]). Solving these problems requires balance between planning in a single mode and planning transitions between modes. Single-mode planning, where each mode defines constraints on the motion of the robot, is discussed in Sec. III-A. Transitioning between modes, e.g., going from not having an object grasped to grasping the object, is discussed in Sec. III-B. Finally, Sec. III-C presents mode families and the multi-modal planning problem.

A. Motion Planning in a Mode

Consider a robot with a configuration space $\mathcal{Q}$. A *mode* imposes *constraints* on a robot's motion. We consider modes defined by *manifold constraints*, shortened to "constraints". See [8] for more on planning under manifold constraints.

Consider a robot in a mode ξ. A mode ξ is defined by a *constraint function* $F^\xi : \mathbb{R}^n \rightarrow \mathbb{R}^{k_\xi}$ $(1 \leq k^\xi < n)$ which is C^2-smooth and is adhered to when $F^\xi(q) = \mathbf{0}$. Planning in a mode requires finding a path in the *mode manifold* $\mathcal{M}^\xi$, an $(n - k^\xi)$-dimensional smooth submanifold of $\mathbb{R}^n$ (Fig. 1a):

$$\mathcal{M}^\xi = \{q \in \mathcal{Q} \mid F^\xi(q) = \mathbf{0}\}.$$

Thus, the single-mode motion planning problem is finding a path from a point $q_{\text{start}} \in \mathcal{M}^\xi$ to some region of interest $Q_{\text{goal}} \subset \mathcal{M}^\xi$ such that the path is collision-free and satisfies mode constraints.

B. Mode Transitions

Transitioning between modes is an essential component of manipulation planning. Consider two modes ξ and ξ' of a robot. To *transition* between ξ and ξ', a path in $\mathcal{M}^\xi$ must be found that ends at a configuration $q' \in \mathcal{M}^\xi \cap \mathcal{M}^{\xi'}$. $\mathcal{M}^\xi \cap$

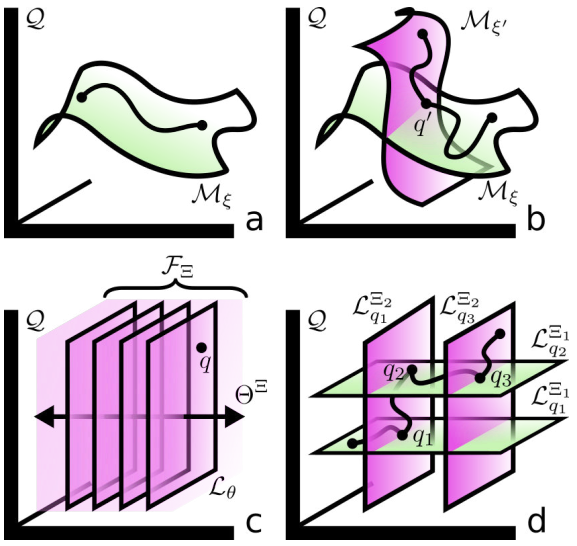

Fig. 1. In **a**, the lower-dimensional manifold $\mathcal{M}^\xi$ in a configuration space $\mathcal{Q}$ is shown. A valid path in mode ξ lies on the surface of this manifold. **b** shows two mode manifolds, $\mathcal{M}^\xi$ and $\mathcal{M}^{\xi'}$, and a path that transitions modes at a transition configuration q'. **c** shows a foliation $\mathcal{F}_\Xi$ for a mode family Ξ, with a transverse manifold Θ_Ξ. Leaves are shown along the transverse, including $\mathcal{L}_\theta$, which is determined by the configuration q. Although shown here as planes, leaf manifolds are not necessarily Euclidean. **d** shows a multi-modal path over two mode families with foliations $\mathcal{F}_{\Xi_1}$ and $\mathcal{F}_{\Xi_2}$. Here, transitions occur at the configurations q_1, q_2, and q_3. Each configuration determines the leaf of the mode family.

$\mathcal{M}^{\xi'}$ is the *transition manifold* between ξ and ξ', with q' being a *transition configuration* (Fig. 1b). Transitions must simultaneously satisfy the constraints of mode ξ and ξ', and thus are an intersection of manifolds, and are of zero volume relative to $\mathcal{M}^\xi$ and $\mathcal{M}^{\xi'}$:

$$\mathcal{M}_{\xi\cap\xi'} = \{q \in \mathcal{Q} \mid F^\xi(q) = \mathbf{0} \wedge F^{\xi'}(q) = \mathbf{0}\}.$$

$\mathcal{M}_{\xi\cap\xi'}$ can also be *empty* if no configuration satisfies both modes (i.e., transitioning is impossible between ξ and ξ'). A key challenge in multi-modal planning is selecting a mode that can be transitioned to, as well as finding transition configurations. To address this challenge, our approach uses discrete "leads" to inform transition selection, described in Sec. IV-D. In our approach, we use projection-based sampling of transition regions to generate candidate transition configurations, which has a non-zero probability of sampling the entire transition manifold [8].

C. Multi-Modal Motion Planning

Manipulation problems are inherently continuous, and thus have a continuous infinity of modes. Consider Ξ, the set of all modes, which can be partitioned into m disjoint *mode families* $\Xi_1, \ldots, \Xi_m$, each of which defines a *foliation*.

DEFINITION III.1: *Foliation [46]*
An n-dimensional manifold $\mathcal{M}$ is a *foliation* if there is a smooth fiber bundle $\mathcal{F}_\mathcal{M} = (\Theta, \mathcal{L}, \pi)$. $\mathcal{F}_\mathcal{M}$ contains a *transverse manifold* Θ of dimension k, a set of disjoint, connected $(n-k)$-dimensional *leaf manifolds* $\mathcal{L}_\theta$ for all $\theta \in \Theta$, and a smooth surjective bundle projection $\pi : \mathcal{M} \to \Theta$. The union of all leaves $\bigcup_{\theta\in\Theta} \mathcal{L}_\theta = \mathcal{M}$.

A mode family is a foliation following their definition by a constraint function F^{Ξ_i} and *co-parameters* $\theta \in \Theta^{\Xi_i}$, where Θ^{Ξ_i} is the transverse manifold of dimension k_{Ξ_i}. Furthermore, we assume the co-parameters are Euclidean, $\Theta^{\Xi_i} \subset \mathbb{R}^{k_{\Xi_i}}$, and can be modeled as $\theta \in [0,1]^{k_{\Xi_i}}$ (this is leveraged in Sec. IV-B for weighting transitions). A *mode* ξ in the mode family is defined by the constraint function and a specific co-parameter θ, $F^{\Xi_i}(q) = \theta$, also written $\langle \Xi_i, \theta \rangle$. A mode corresponds to a leaf manifold $\mathcal{L}_\theta$:

$$\mathcal{M}^\xi = \mathcal{L}_\theta = \{q \in \mathcal{Q} \mid F^{\Xi_i}(q) = \theta\}.$$

Defining mode families as foliations puts manifold constraints and the definition of modes under one umbrella, enabling use of the general constrained planning framework presented in [8]. An example of a foliation is shown in Fig. 1c.

Mode families are intuitively the "classes" of modes. For example, a mode family could correspond to all placements of an object on a table: the object could be placed anywhere on the table, the co-parameter in this case being a two-dimensional coordinate of the objects placement on the table, defining the particular mode. *Co-parameters* are essential to the efficiency of our method—they enable us to succinctly encode complicated geometric state of the problem.

Note, transitions are only possible between different mode families, and not within the mode family itself. For example, a robot can typically not instantaneously change its grasp; it will have to perform a regrasp, which corresponds to at least two mode transitions. The multi-modal planning problem is thus to find a continuous sequence of collision-free paths, each of which is a valid path in some mode ending in a valid transition to the subsequent mode. An example multi-modal path is shown in Fig. 1d.

IV. Our Multi-Modal Motion Planner

Our algorithm is able to handle problems that interleave complex discrete structures with difficult continuous single-mode planning. This discrete structure (e.g., information about which objects are grasped by which grippers) is captured by a *mode transition graph* which we augment with information from single-mode planning (Sec. IV-B). Based on this graph, we compute *leads*—promising sequences of modes to reach the goal (Sec. IV-D). Leads are used to guide the search of a lower-level motion planner, which then informs edge weights (Sec. IV-C). We combine these layers of planning synergistically to better inform the overall search.

A. Baseline Method

Our algorithm builds on concepts from [1], which plans given a similar model of a multi-modal problem. The algorithm presented in [1] transitions between modes by choosing the next mode using an RRT-like extension scheme, extending at random. Our approach modifies the extension step by informing search with a discrete lead, described in Sec. IV-D. Additionally, [1] uses a "utility-centered expansion strategy,"

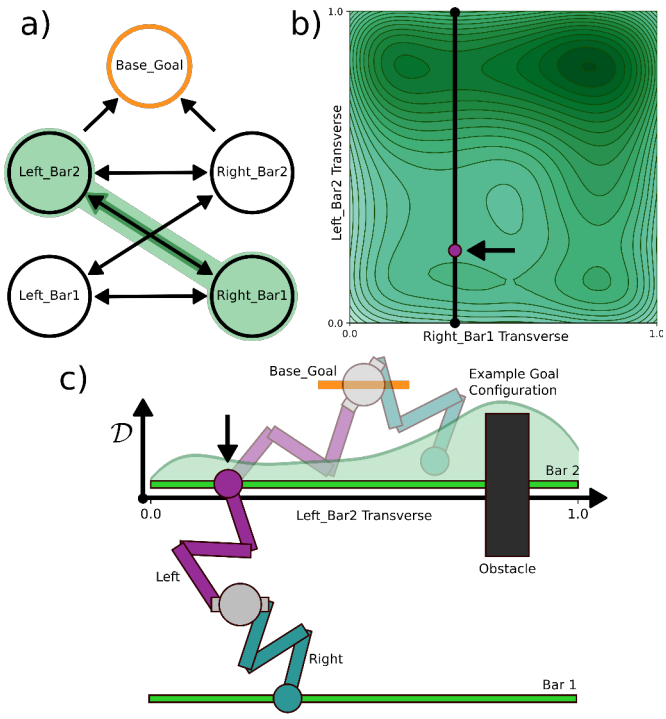

Fig. 2. Visualization of transition graph weights. **a)** A transition graph for the "simple monkey" domain. The transition from grasping bar 1 with the right limb ("Right_Bar1") to grasping bar 2 with the left limb ("Left_Bar2") is highlighted. **b)** A contour map of the weights for the highlighted transition (gathered offline). The co-parameter for grasping bar 1 with the right limb is plotted on the X-axis, while the co-parameter of grasping bar 2 with the left limb is plotted on the Y-axis. Darker shading represent higher-weight (transitions we expect to be difficult). A slice for a specific X-axis co-parameter (i.e., initial placement of the right gripper) is marked in black, and shown in **c)**. **c)** The "simple monkey" domain, with the weights of the slice from **b)** visualized over bar 2. The left arm of the robot is shown grasping bar 2 (highlighted by the arrow), and thus is a transition configuration. The transition is marked at its co-parameters by a purple point in **b)**. The robot must grasp bar 2 to eventually reach "Base_Goal", visualized as an orange region for the base and an example satisfying configuration.

where promising transitions are taken based on a precomputed "utility." Our approach also approximates how likely a transition is to succeed, but does so online (Sec. IV-C). This is similar to the weighting done in [4], but for continuously infinite modes.

Moreover, we use a very general formulation of mode families: a constraint function F^{Ξ} (recall Sec. III). For example, in Fig. 2c, the constraint that defines the mode family for grasping the handrail is given as a function of the robot's kinematics, requiring the end-effector to be positioned along the rail. We leverage the general constrained sampling-based planning framework described in [8] to enable single-mode planning given general mode constraints (using a manifold-constrained PRM, which is reused). Mode transitions are sampled using projection-based sampling over the intersection of the leaf manifold and destination manifold.

B. Mode Transition Graphs

Key to multi-modal planning is transitioning between modes. Recall from Sec. III-C there exist a set of mode families $\Xi_1, \ldots, \Xi_m$. In addition to it being impossible to transition between modes within a family, there are potentially mode families that cannot transition to each other. Consider a pick-and-place domain with two objects. One encoding has mode families for picking up either object: Ξ_{obj1} and Ξ_{obj2}. But, the robot's gripper can only pick up one object at a time.

We encapsulate this encoding as a *mode transition graph*, similar to [1]. A mode transition graph $\mathcal{G}$ is composed of vertices $V = \Xi_1, \ldots, \Xi_m$, where each vertex corresponds to a mode family. There exist directed edges $\langle \Xi_i, \Xi_j \rangle \in E \subseteq V \times V$ that denote possible valid directed transitions between mode families (shown in Fig. 2a).

In addition, the transition graph contains statistics on what transitions are likely to succeed. Attached to every directed edge $\langle \Xi_{\text{src}}, \Xi_{\text{dst}} \rangle$ are *transition weights* $\mathcal{D}^{\Xi_{\text{src}}, \Xi_{\text{dst}}}(\theta_{\text{src}}, \theta_{\text{dst}})$. The transition weights $\mathcal{D}^{\Xi_{\text{src}}, \Xi_{\text{dst}}}(\theta_{\text{src}}, \theta_{\text{dst}})$ are a distribution with support over the transverse manifolds for the source and destination mode families,

$$\mathcal{D}^{\Xi_{\text{src}}, \Xi_{\text{dst}}}(\theta_{\text{src}}, \theta_{\text{dst}}) : \Theta^{\Xi_{\text{src}}} \times \Theta^{\Xi_{\text{dst}}} \to \mathbb{R}.$$

Informally, $\mathcal{D}^{\Xi_{\text{src}}, \Xi_{\text{dst}}}(\theta_{\text{src}}, \theta_{\text{dst}})$ captures the difficulty of the single-mode motion planning problem of transitioning to the mode $\xi_{\text{dst}} \in \Xi_{\text{dst}}$ (where ξ_{dst} is the mode determined by the co-parameter $\theta_{\text{dst}} \in \Theta^{\Xi_{\text{dst}}}$) to the mode $\xi_{\text{src}} \in \Xi_{\text{src}}$ (where ξ_{src} is the mode determined by the co-parameter $\theta_{\text{src}} \in \Theta^{\Xi_{\text{src}}}$). Specifically the weights are inversely proportional to the probability of single-mode motion planning to succeed within a fixed time budget on the leaf manifold $\mathcal{L}_{\theta_{\text{src}}}$, $\theta_{\text{src}} \in \Theta^{\Xi_{\text{src}}}$ to any transition configuration that may or may not exist on the leaf manifold $\mathcal{L}_{\theta_{\text{dst}}}$, $\theta_{\text{dst}} \in \Theta^{\Xi_{\text{dst}}}$.

Recall from Def. III.1 that the transverse manifolds for a foliation are Euclidean. The transverse weights are maintained over a Euclidean hypercube, $[0, 1]^{k_{\Xi_{\text{src}}} + k_{\Xi_{\text{dst}}}}$. An example of these weights is visualized in Fig. 2b. However, these distributions are not known *a priori*. Estimating these distributions online is key to effective multi-modal planning. We cover estimating these distributions in Sec. IV-C.

Consider the example shown in Fig. 2. A robot with two end-effectors that can climb by grasping bars must position its base in a goal region above bar 2 ("Base_Goal"). The graph contains all possible transitions; there is no transition from "{Left, Right}_Bar1" to "Base_Goal" because it is out of reach, and none between the same end-effector on a bar as the robot must always be grasping a bar. The obstacle on one side of the bar makes movement more difficult, corresponding to a higher weight (seen in the distribution in Fig. 2b and c). Intuitively, this shows that if the robot is trying to transition from "Right_Bar1" at the grasped location to "Left_Bar2", then it is important to consider where it grasps Bar2. Note that the weights for Fig. 2b were generated by many offline runs, while our experiments approximate this distribution online without precomputation.

C. Informing the Transition Graph

As mentioned, the transition weights are meant to capture the likelyhood that single-mode planning will be successful in moving from one ⟨mode family, co-parameter⟩ pair to the next. We update the weights using a simple weighting scheme,

based on information gathered during the search. There are three events that we can use to update the information, given a single-mode planning instance of $\langle \Xi_{\text{src}}, \theta_{\text{src}} \rangle$ to $\langle \Xi_{\text{dst}}, \theta_{\text{dst}} \rangle$.

- *Planning is Successful*: We add a small penalty to the weight to encourage exploration of alternate routes.
- *Planning is Unsuccessful*: We add a larger penalty to the weight to discourage attempting this transition again.
- *No Transition Found*: That is, no transition configuration was sampled from $\langle \Xi_{\text{src}}, \theta_{\text{src}} \rangle$ to $\langle \Xi_{\text{dst}}, \theta_{\text{dst}} \rangle$. We add the largest penalty to the weight.

Moreover, we add the penalty to all "nearby" co-parameters. To distribute the weight over nearby co-parameters, the penalty is applied to all adjacent co-parameter pairs with an exponential drop off. Thus, search is biased not only against attempting this transition again, but also nearby "similar" transitions, as nearby co-parameter pairs will have similar planning conditions due to continuity. We conjecture this weighting scheme maintains probabilistic completeness as we never rule out a possible transition.

Our experiments show that this simple weighting scheme can dramatically improve runtime on a variety of scenes. We used fixed weights for all experiments of 3, 5, and 10 respectively for the three weights. Other weighting approaches are possible, but this simple scheme gave us excellent results.

D. Building Leads

Given a mode family transition graph with suitable weights, we compute leads to bias our search. Essentially, the idea of a lead is to suggest the most likely sequence of mode transitions to reach a destination. In our algorithm, a destination is a sample at random or from the goal, and the source is an existing configuration in the search tree. Formally, we have:

DEFINITION IV.1: *Augmented Shortest Path Problem*
Given a transition graph $\mathcal{G}$, a starting mode specified by a mode family and co-parameter $\langle \Xi_{\text{src}}, \theta_{\text{src}} \rangle$, and destination mode $\langle \Xi_{\text{dst}}, \theta_{\text{dst}} \rangle$, find the lowest cumulative weight path, $\langle \Xi_1, \theta_1 \rangle \dots \langle \Xi_k, \theta_k \rangle$, that minimizes the total cost:

$$\sum_{i=1}^{k-1} \mathcal{D}^{\Xi_i, \Xi_{i+1}}(\theta_i, \theta_{i+1})$$

Thus, a lead is a sequence of ⟨mode family, co-parameter⟩ pairs, realized in sequence by single-mode planning. In practice, we do not produce enough samples to make exact computations of this shortest path worthwhile. Instead, we approximate the continuous distribution by discretizing the co-parameter for each mode family into intervals, tracking weights over transitions from interval to interval. We use Dijkstra's shortest path algorithm over the discretized mode family transition graph to solve Def. IV.1, generating a lead.

In an iteration of planning, a random configuration and mode or goal configuration is sampled with some bias. Using an RRT-like scheme, a node from the existing tree is selected to expand from. A lead is generated from this node to the sample, guiding single-mode search to extend the search tree.

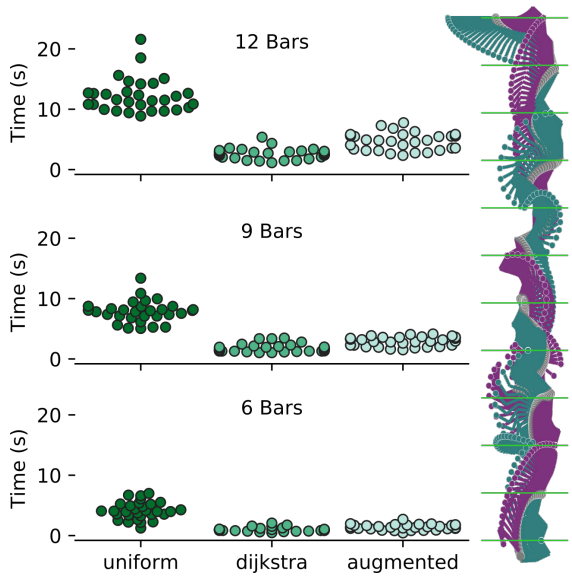

Fig. 3. Timing results for three "long monkey" domains. On the right the "12 Bars" version of the environment is shown, along with the swept volume of by an example multi-modal plan. On the left, timing results for "6 Bars," "9 Bars," and "12 Bars" are shown for the uniform, Dijkstra, and augmented methods. Each dot corresponds one of the 30 trials done for each scenario and planner. Note that as the problems become more challenging (i.e., there are more bars and thus a longer sequence of transitions needed), the benefits of using the discrete leads become more pronounced. In this scenario the extra computation and specificity of the augmented method has no benefits over the Dijkstra-based method, but the overhead is not significant.

V. EXPERIMENTS

Our experiments are chosen to measure the importance of the discrete component (building leads) and continuous component (updating transition weights) of our algorithm in various scenarios. We compare three algorithms:

1) *Uniform*, which chooses mode transitions uniformly at random from the neighboring modes (essentially, an emulation of [1] by our planner).
2) *Dijkstra*, which searches for a sequence of mode family transitions, but does not select suitable co-parameters.
3) *Augmented*, our proposed method which searches for a sequence of mode family transitions as well as suitable co-parameters.

Uniform is the baseline. Many manipulation problems have challenging discrete structure, in that many transitions are necessary to achieve the task. Intuitively, *Dijkstra* should perform well when this discrete structure is important, because good choices of which mode to transition to make the search much more efficient (e.g., Fig. 3). *Augmented* should perform well relative to *Dijkstra* when the continuous co-parameters are relatively important (e.g., Fig. 4). Using the augmented transition graph also allows us solve difficult manipulation problems (see Fig. 5) more efficiently.

Fig. 3 shows results for the "long monkey" domains. Here, a "monkey" robot with two end-effectors grasps handrails to move through the environment (similar to Fig. 2). The "monkey" has nine degrees of freedom (three for each arm and three for the pose of the base). There are two mode families for each bar, corresponding to grasping the bar

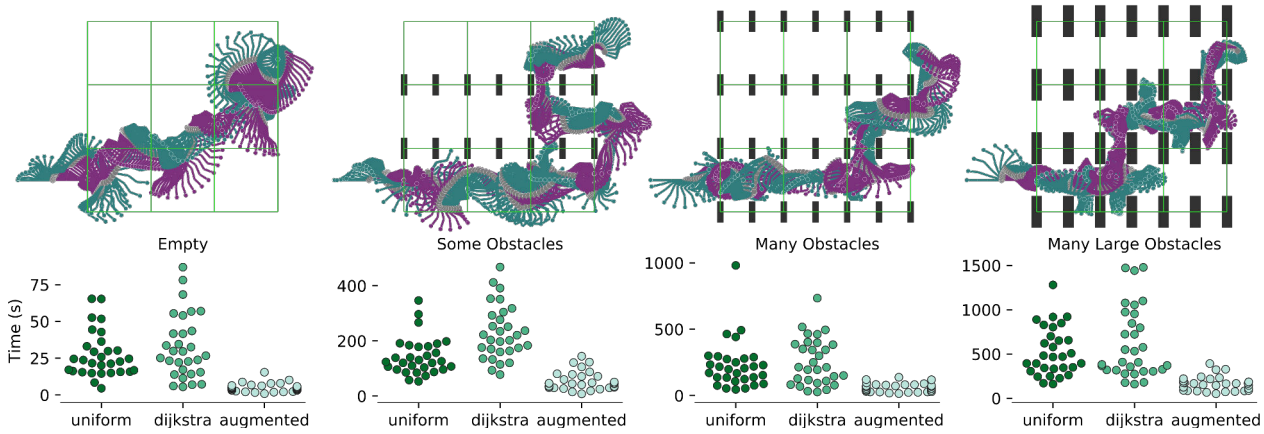

Fig. 4. Timing results for "lateral monkey" domains. On top, the swept volume of an example multi-modal plan is shown for each of the domains. Obstacles are shown as dark rectangles, and the graspable handrails are shown in green. On bottom, timing results for each environment are shown: each dot corresponds one of the 30 trials done for each scenario and planner. As the problems become more challenging (i.e., the clutter in the scene increases), the benefits of the augmented heuristic become more pronounced. This happens because as the weighting scheme helps bias away from known failure regions and toward unexplored transitions. Note the Y-axis changes between each plot.

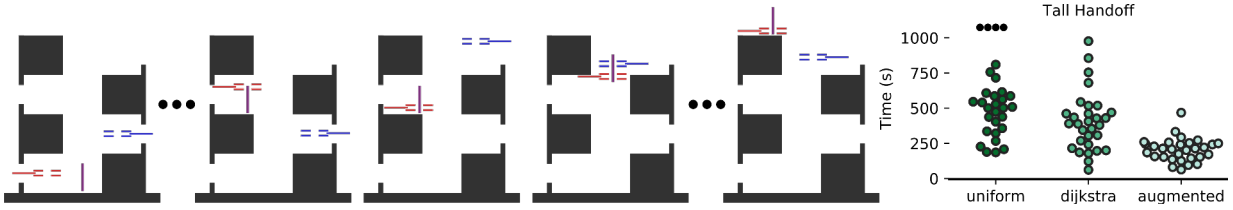

Fig. 5. Timing results for the "tall handoff" domain. On the left, five stills are shown from an example generated multi-modal plan in this domain. The start and goal are shown on the far left and right respectively. The end-effectors are red and blue, the graspable object is purple. An example of a necessary regrasp to achieve a handoff is shown in the middle. Timing results are shown on the right, 30 trials for each planner. In difficult TMP-like scenarios, the augmented heuristic makes our search more efficient. The *Uniform* method experienced four failures, shown as black dots.

with either end-effector—the co-parameter corresponds to the location grasped on the bar. Fig. 3 shows that *Dijkstra*'s works well when the discrete aspect of the problem is the primary challenge. Intuitively this makes sense as the robot must traverse many bars—choosing a good order makes the problem significantly easier. *Dijkstra*'s helps the robot choose *which* rung to grasp.

Fig. 4 shows results in the "lateral monkey" domains, each domain increasing in clutter. There are eight bars the monkey can grasp in each domain, with the same "monkey" robot as before. Fig. 4 shows that we improve significantly over *Dijkstra*'s in problems where choosing the right the co-parameters becomes a significant aspect of the problem. While similar to the above problem, the bars are much longer; the greater latitude means *where* the robot grasps the bar (the co-parameter) is much more important in this problem.

Fig. 5 shows results for the "tall handoff" domain. In this case, there are two translating end-effectors that can grasp a long object. Here, there are seven mode families: five corresponding to the placement of the object along the flat surfaces, and a mode family corresponding to each end-effector grasping the object—the co-parameter corresponds to where the end-effector grasps the object. Fig. 5 shows that we can solve complex TMP-like problems. This scenario is difficult due to obstacles preventing the object from being removed from the narrow passage while grasped, necessitating a sequence of handoffs to reach the goal. The robot needs to repeatedly place the object in order to regrasp the object, so it may be handed off. This combines a difficult motion planning problem with the discrete structure inherent to the problem. Our approach can solve problems intractable in typical TMP frameworks (e.g., [23]), as search looks for short tasks plans. Such plans are not feasible, so these methods waste time by assuming the "best" plan has fewer actions, while our approach inherently biases towards solving the motion planning problem using discrete structure as a guide.

VI. Discussion

We have presented a general layered multi-modal planning framework to solve manipulation planning problems. Our general formulation enables us to approach and inform search across a wide variety of problems with continuous infinities of modes. Moreover, we showed that both discrete leads and informing discrete search with continuous planning yields more efficient planning which can scale to complex problems.

Currently, our method provides feasible multi-modal motion, but not "optimal" motion with respect some cost (e.g., minimizing the number of transitions, total path length). Similar to optimizing heursitics for sampling-based planners [47], optimizing multi-modal paths is a fruitful line of future work, similar to the heuristics used in [6] or joint optimization [41]–[43]. In the future, we also plan to extend our work to more complicated scenes such as robots with 3D workspaces and more complicated mode transition graphs, similar to task and motion planning domains.

REFERENCES

[1] K. Hauser and V. Ng-Thow-Hing, "Randomized multi-modal motion planning for a humanoid robot manipulation task," *Int. J. of Robotics Research*, vol. 30, no. 6, pp. 678–698, 2011.

[2] J. Barry, L. P. Kaelbling, and T. Lozano-Pérez, "A hierarchical approach to manipulation with diverse actions," in *IEEE Int. Conf. Robot. Autom.* IEEE, 2013, pp. 1799–1806.

[3] W. Vega-Brown and N. Roy, "Asymptotically optimal planning under piecewise-analytic constraints," in *Int. Wksp. on the Algorithmic Foundations of Robotics*. Springer, 2016.

[4] C. L. Nielsen and L. E. Kavraki, "A two level fuzzy PRM for manipulation planning," in *IEEE/RSJ Int. Conf. Intell. Robot. and Syst.*, 2000, pp. 1716–1721.

[5] T. Siméon, J.-C. Laumond, J. Cortés, and A. Sahbani, "Manipulation planning with probabilistic roadmaps," *Int. J. of Robotics Research*, vol. 32, no. 7-8, pp. 729–746, 2004.

[6] S. Cambon, R. Alami, and F. Gravot, "A hybrid approach to intricate motion, manipulation and task planning," *Int. J. of Robotics Research*, vol. 28, no. 1, pp. 104–126, 2009.

[7] E. Plaku, L. E. Kavraki, and M. Y. Vardi, "Motion planning with dynamics by a synergistic combination of layers of planning," *IEEE Trans. Robot.*, vol. 26, no. 3, pp. 469–482, 2010 2010.

[8] Z. Kingston, M. Moll, and L. E. Kavraki, "Exploring implicit spaces for constrained sampling-based planning," *Int. J. of Robotics Research*, vol. 38, no. 10–11, pp. 1151–1178, 2019.

[9] J. J. Kuffner and J. Xiao, *Springer Handbook of Robotics*, 2nd ed. Springer, 2016, ch. Motion for Manipulation Tasks, pp. 867–896.

[10] M. T. Mason, "Toward robotic manipulation," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, no. 1, pp. 1–28, 2018.

[11] R. Alami, J.-P. Laumond, and T. Siméon, "Two manipulation planning algorithms," in *Int. Wksp. on the Algorithmic Foundations of Robotics*, 1994, pp. 109–125.

[12] Y. Koga and J.-C. Latombe, "On multi-arm manipulation planning," in *IEEE Int. Conf. Robot. Autom.*, 1994, pp. 945–952.

[13] J. Barraquand and P. Ferbach, "A penalty function method for constrained motion planning," in *IEEE Int. Conf. Robot. Autom.*, 1994.

[14] P. Ferbach and J. Barraquand, "A method of progressive constraints for manipulation planning," *IEEE Trans. Robot. Autom.*, no. 4, pp. 473–485, 1997.

[15] H. M. Choset, S. Hutchinson, K. M. Lynch, G. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun, *Principles of Robot Motion: Theory, Algorithms, and Implementation*. MIT Press, 2005.

[16] S. M. LaValle, *Planning Algorithms*. Cambridge University Press, 2006.

[17] L. E. Kavraki and S. M. LaValle, *Springer Handbook of Robotics*, 2nd ed. Springer, 2016, ch. Motion Planning, pp. 139–162.

[18] M. Zucker, N. Ratliff, A. D. Dragan, M. Pivtoraiko, M. Klingensmith, C. M. Dellin, J. A. Bagnell, and S. S. Srinivasa, "CHOMP: Covariant hamiltonian optimization for motion planning," *Int. J. of Robotics Research*, vol. 32, no. 9-10, pp. 1164–1193, 2013.

[19] J. Schulman, Y. Duan, J. Ho, A. Lee, I. Awwal, H. Bradlow, J. Pan, S. Patil, K. Goldberg, and P. Abbeel, "Motion planning with sequential convex optimization and convex collision checking," *Int. J. of Robotics Research*, vol. 33, no. 9, pp. 1251–1270, 2014.

[20] D. Berenson, S. S. Srinivasa, and J. J. Kuffner, "Task space regions: A framework for pose-constrained manipulation planning," *Int. J. of Robotics Research*, vol. 30, no. 12, pp. 1435–1460, Oct. 2011.

[21] Z. Kingston, M. Moll, and L. E. Kavraki, "Sampling-based methods for motion planning with constraints," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, no. 1, pp. 159–185, 2018.

[22] E. Vidal Garcia, M. Moll, N. Palomeras, J. D. Hernández, M. Carreras, and L. E. Kavraki, "Online multilayered motion planning with dynamic constraints for autonomous underwater vehicles," in *IEEE Int. Conf. Robot. Autom.*, May 2019, pp. 8936–8942.

[23] N. T. Dantam, Z. Kingston, S. Chaudhuri, and L. E. Kavraki, "An incremental constraint-based framework for task and motion planning," *Int. J. of Robotics Research*, vol. 37, no. 10, pp. 1134–1151, 2018.

[24] K. Hauser, T. Bretl, J.-C. Latombe, K. Harada, and B. Wilcox, "Motion planning for legged robots on varied terrain," *Int. J. of Robotics Research*, vol. 27, no. 11-12, pp. 1325–1349, 2008.

[25] T. Bretl, "Motion planning of multi-limbed robots subject to equilibrium constraints: The free-climbing robot problem," *Int. J. of Robotics Research*, vol. 25, no. 4, pp. 317–342, 2006.

[26] J. Mirabel, S. Tonneau, P. Fernbach, A.-K. Seppälä, M. Campana, N. Mansard, and F. Lamiraux, "HPP: A new software for constrained motion planning," in *IEEE/RSJ Int. Conf. Intell. Robot. and Syst.*, 2016, pp. 383–389.

[27] J. Mirabel and F. Lamiraux, "Manipulation planning: addressing the crossed foliation issue," in *IEEE Int. Conf. Robot. Autom.*, 2017, pp. 4032–4037.

[28] P. S. Schmitt, F. Wirnshofer, K. M. Wurm, G. von Wichert, and W. Burgard, "Modeling and planning manipulation in dynamic environments," in *IEEE Int. Conf. Robot. Autom.*, 2019, pp. 176–182.

[29] B. Burns and O. Brock, "Toward optimal configuration space sampling," in *Robotics: Science and Syst.*, 2005, pp. 105–112.

[30] J. Pan, S. Chitta, and D. Manocha, "Faster sample-based motion planning using instance-based learning," in *Int. Wksp. on the Algorithmic Foundations of Robotics*, E. Frazzoli, T. Lozano-Perez, N. Roy, and D. Rus, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 381–396.

[31] T. Siméon, J. Cortés, A. Sahbani, and J.-P. Laumond, "A manipulation planner for pick and place operations under continuous grasps and placements," in *IEEE Int. Conf. Robot. Autom.*, vol. 2, May 2002, pp. 2022–2027.

[32] A. Sahbani, J. Cortés, and T. Siméon, "A probabilistic algorithm for manipulation planning under continuous grasps and placements," in *IEEE/RSJ Int. Conf. Intell. Robot. and Syst.*, vol. 2, Sept. 2002, pp. 1560–1565.

[33] S. Cambon, F. Gravot, and R. Alami, "A robot task planner that merges symbolic and geometric reasoning," in *European Conference on Artifical Intelligence*. IOS Press, 2004, pp. 895–899.

[34] F. Gravot, S. Cambon, and R. Alami, "aSyMov: a planner that deals with intricate symbolic and geometric problems," in *Int. Symp. on Robotics Research*. Springer, 2005, pp. 100–110.

[35] J. Van Den Berg, M. Stilman, J. Kuffner, M. Lin, and D. Manocha, "Path planning among movable obstacles: a probabilistically complete approach," in *Algorithmic Foundation of Robotics VIII*. Springer, 2009, pp. 599–614.

[36] J. Barry, "Manipulation with diverse actions," Ph.D. dissertation, Massachusetts Institute of Technology, June 2013.

[37] J. Barry, K. Hsiao, L. P. Kaelbling, and T. Lozano-Pérez, "Manipulation with multiple action types," in *Experimental Robotics*. Springer, 2013, pp. 531–545.

[38] L. P. Kaelbling and T. Lozano-Pérez, "Hierarchical task and motion planning in the now," in *IEEE Int. Conf. Robot. Autom.*, 2011, pp. 1470–1477.

[39] S. Srivastava, E. Fang, L. Riano, R. Chitnis, S. Russell, and P. Abbeel, "Combined task and motion planning through an extensible planner-independent interface layer," in *IEEE Int. Conf. Robot. Autom.*, 2014, pp. 639–646.

[40] C. R. Garrett, T. Lozano-Pérez, and L. P. Kaelbling, "Sampling-based methods for factored task and motion planning," *Int. J. of Robotics Research*, vol. 37, no. 13–14, pp. 1796–1825, 2018.

[41] M. Toussaint, "Logic-geometric programming: An optimization-based approach to combined task and motion planning," in *Int. Joint Conf. on Artificial Intell.*, 2015, pp. 1930–1936.

[42] M. Toussaint, K. R. Allen, K. A. Smith, and J. B. Tenenbaum, "Differentiable physics and stable modes for tool-use and manipulation planning," in *Robotics: Science and Syst.*, 2018.

[43] C. Phiquepal and M. Toussaint, "Combined task and motion planning under partial observability: An optimization-based approach," in *IEEE Int. Conf. Robot. Autom.*, 2019, pp. 9000–9006.

[44] K. He, M. Lahijanian, L. E. Kavraki, and M. Y. Vardi, "Towards manipulation planning with temporal logic specifications," in *IEEE Int. Conf. Robot. Autom.*, 2015, pp. 346–352.

[45] A. Bhatia, M. Maly, L. E. Kavraki, and M. Y. Vardi, "Motion planning with complex goals," *IEEE Robot. Autom. Magazine*, vol. 18, no. 3, pp. 55–64, 2011.

[46] M. Spivak, *A Comprehensive Introduction to Differential Geometry*. Publish or Perish, 1999.

[47] R. J. Geraerts and M. H. Overmars, "Creating high-quality paths for motion planning," *Int. J. of Robotics Research*, vol. 26, no. 8, pp. 845–863, 2007.