To appear in the Intl. J. of Robotics Research, 2020, http://dx.doi.org/10.1177/0278364919890408

A Scalable Motion Planner for High-Dimensional Kinematic Systems

Ryan Luna, Mark Moll, Julia Badger, and Lydia E. Kavraki

Abstract

Sampling-based algorithms are known for their ability to effectively compute paths for high-dimensional robots in relatively short times. The same algorithms, however, are also notorious for poor quality solution paths, particularly as the dimensionality of the system grows. This work proposes a new probabilistically complete sampling-based algorithm, XXL, specially designed to plan the motions of high-dimensional mobile manipulators and related platforms. Using a novel sampling and connection strategy that guides a set of points mapped on the robot through the workspace, XXL scales to realistic manipulator platforms with dozens of joints by focusing the search of the robot's configuration space to specific degrees-of-freedom that affect motion in particular portions of the workspace. Simulated planning scenarios with the Robonaut2 platform and planar kinematic chains confirm that XXL exhibits competitive solution times relative to many existing works while obtaining execution-quality solution paths. Solutions from XXL are of comparable quality to cost-aware methods even though XXL does not explicitly optimize over any particular criteria, and are computed in an order of magnitude less time. Furthermore, observations about the performance of sampling-based algorithms on high-dimensional manipulator planning problems are presented that reveal a cautionary tale regarding two popular guiding heuristics used in these algorithms, indicating that a nearly random search may outperform the state-of-the-art when defining such heuristics is known to be difficult.

1 Introduction

Motion planning is an essential tool for developing robotic systems that operate with any level of autonomy. Mobile manipulation platforms are being constructed with heterogeneous sensor suites, dozens of actuators, and onboard computing resources to support increasingly complex missions. Moving such a robot requires careful planning and deliberation to coordinate the actions of an increasing number of actuators. Automating the motion planning process, however, is known to be computationally difficult even for simple systems with relatively few degrees-of-freedom (DoFs).

The primary focus of this work is the reliable computation of collision-free paths considering only geometric (e.g., kinematic) constraints that bring a high-DoF manipulator to a pre-grasp pose in a static, possibly confined environment. Inspiration is drawn from the Robonaut2 (R2) platform, a humanoid robot with seven-DoF arms and legs, dexterous hands, and broad sensing capabilities, developed at NASA and deployed on the International Space Station (ISS) (Diftler et al., 2011, 2012; Badger et al., 2013). R2 navigates the ISS by grasping special handrails attached to the interior of each module with grippers on its feet, as shown in Figure 1a. Planning with only geometric constraints may sound simplistic, but solutions to such instances are realizable in microgravity environments like R2 aboard the ISS (Diftler et al., 2011), or cases where the robot moves with sufficiently small velocities to minimize dynamical effects (Hauser et al., 2008).

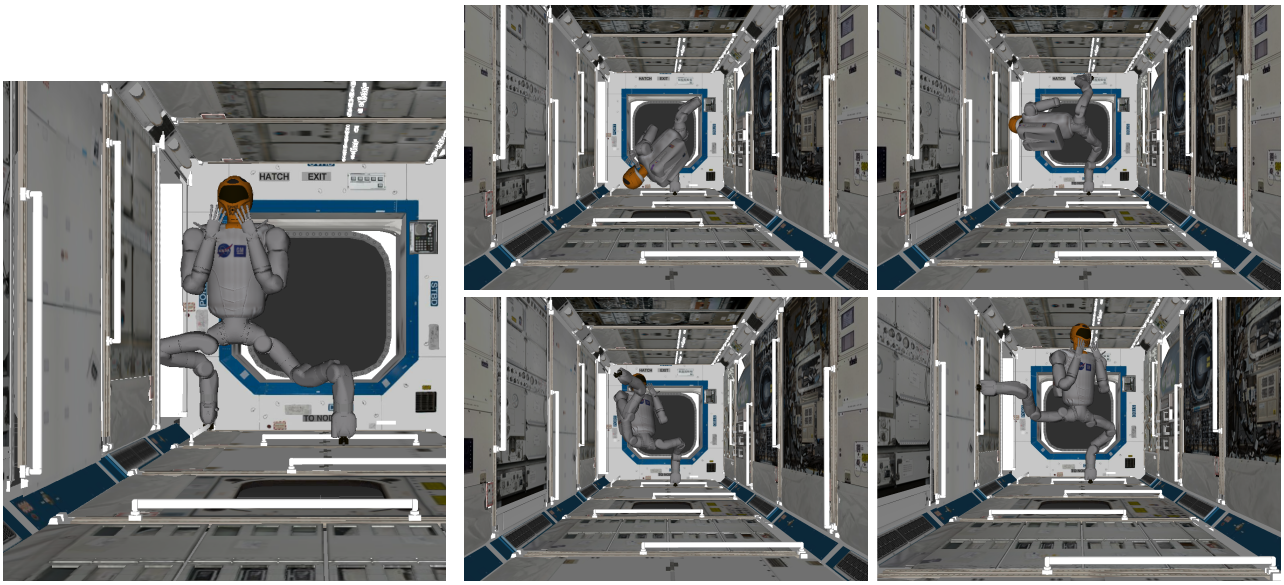
There are several well-studied techniques for planning the motions of high-DoF robots like R2. One popular approach is to translate the planning problem into a set of constraints and use optimization techniques to find a valid solution path (e.g., Quinlan and Khatib, 1993; Toussaint, 2009; Kalakrishnan et al., 2011; Kobilarov, 2011; Zucker et al., 2013; Schulman et al., 2014). Optimization-based approaches typically deform a global path in a local manner to satisfy all of the constraints. These

techniques excel in high-dimensional configuration spaces and are able to directly reason over non-holonomic and other high-order constraints, often converging swiftly to a high-quality solution that lies in a local minimum of the cost landscape. Due to the inherent greediness of the optimization process, however, optimization-based approaches are only locally optimal. In the worst case, these methods can get trapped in local minima containing invalid paths and may fail to return a valid solution. Thus, optimization-based planning methods do not generally provide completeness guarantees. For motion planning queries that require a relatively long solution path, greedy approaches may not be effective at finding solution for systems like R2 unless the input path already lies near a local minimum. Further, it is difficult to generalize and deploy an optimization-based approach without reconfiguring constraints and tuning cost parameters for the specific system. On R2, optimization methods for gradient descent are challenging for two-legged motions due to the large number of singular joint configurations. When following the Jacobian to guide the legs through the workspace, for example, it is common to reach a configuration at the angular limit of one or more joints for all but the shortest of motions. Further compounding this problem is the confined space that the robot operates in. Small perturbations of a few leg joints can easily result in an colliding configuration for R2 in the ISS. The structure of the problem itself must usually be encoded in the form of cost or constraints. Moreover, this set of constraints changes dynamically with the fixed base(s) of the robot.

On the other end of the planning spectrum are techniques that exhaustively search the space of valid robot configurations. Since the search space for most robot motion planning problems is continuous, these methods implicitly discretize the space into a lattice of motions that can be searched using standard graph search techniques (e.g., Likhachev and Ferguson, 2009). Unlike optimization-based approaches, searching the space of robot configurations generally admits completeness and optimality guarantees up to the discretization resolution of the underlying lattice; searching the space of joint configurations makes these methods more robust to joint limits compared to their optimization counterparts. If there exists a narrow valid region of the configuration space that must be explored, however, search-based methods may be unable to find a solution when the narrow region is smaller than the discretization resolution. Efficiently searching a lattice for a high-DoF robot, however, poses a difficult computational challenge that often requires expertly-crafted, system-specific dimensionality reduction techniques to compactly encode the planning problem in a tractable form. Recent work in this area plans for a 14-DoF dual-arm manipulator, but relies heavily on a closed kinematic chain assumption to reduce the dimensionality of the search space (Cohen et al., 2014).

Sampling-based algorithms are another type of approach that estimate the connectivity of the valid search space by sampling valid robot configurations and connecting the configurations with short valid motions (e.g., Kavraki et al., 1996; Hsu et al., 1999; LaValle and Kuffner, 2001). These techniques excel in high-dimensional search spaces since a relatively sparse set of sampled configurations may be necessary to construct a valid solution path. The sampling process, however, results in notoriously sub-optimal solutions that require post-processing and/or subsequent optimization (e.g., Geraerts and Overmars, 2007) before they are executed by the robot. In practice, post-processing methods can quickly and dramatically improve the quality of a path but are still susceptible to local minima and are unlikely to arrive at a solution that lies in a different minimum or homotopy class. On the other hand, the sampling process also typically admits a weaker *probabilistic completeness* guarantee that ensures a solution is eventually found when one exists, making sampling-based methods a good candidate for further investigation. Solution paths from these methods also complement optimization-based techniques that often depend on a good initial path.

Unsurprisingly, planning the motions of R2 using a canonical sampling-based algorithm often results in paths with undesirable characteristics and an overall appearance of superfluity. Figure 1a shows R2 inside the Destiny module of the ISS, ready to begin executing a task. Planning the motions from handrail to handrail using existing methods often produces unexpected results. In the worst case, these algorithms require R2's torso to perform a somersaulting motion even after applying standard



(a) An initial pose for R2 in the Destiny module.

(b) R2 moving its right foot to grasp a handrail on the wall of the International Space Station. This path was obtained using existing algorithms implemented in the *Open Motion Planning Library* (Şucan et al., 2012).

Figure 1: Applying standard sampling-based methods when moving Robonaut2's legs often results in somersaulting or flipping motions of the torso.

post-processing techniques to improve the quality of paths generated by sampling-based methods. Such a motion, shown in Figure 1b, is highly undesirable since the somersault is unintuitive and unpredictable to nearby astronauts and the momentum generated by the torso mass is likely to exceed safety tolerances established for the space station environment. The somersault is a direct result of the *local connector* employed in many sampling-based algorithms that connects configurations along a straight-line in the configuration space. A straight path through R2's configuration space minimizes the absolute change in joint angles, but the planner has no knowledge that even small angular changes in R2's hip joint can cause large and often undesirable motions in the torso. More generally, a short path through the configuration space of a high-DoF manipulator does not imply a short path through the workspace. Choosing which configurations to connect and the strategy employed to connect them are the most fundamental decisions in the design of a sampling-based planner since these primitives strongly influence both the theoretical and the practical properties of the resulting algorithm. Simple schemes to improve the quality of the resulting path, such as artificially limiting the joint angles or planning directly in the robot's workspace, come with their own set of complications: planning in the workspace is inherently incomplete and artificially limiting joint angles may prune away all solutions to an otherwise feasible problem.

This work introduces XXL, a probabilistically complete sampling-based algorithm specifically designed to plan the motions of high-DoF manipulator platforms like R2. XXL coordinates a search through the robot's configuration space using a decomposition of the robot's workspace in conjunction with a novel workspace-guided sampling and connection strategy that takes advantage of the underlying kinematic constraints of the robot to aggressively move proximal links to a desired workspace region before shifting the focus to distal links. To achieve scalability to high-DoF manipulators, XXL restricts the search to a subset of joints that affect motions in promising areas of the workspace at any given time. Although solution optimization is not a direct objective of this work, XXL often arrives at solution paths that do not exhibit the superfluity as in Figure 1. These paths are beneficial not

only for R2 operating autonomously onboard the ISS, but in a much broader class of situations where robots operate near or even with people where the paths executed by the robot must be intuitive (Sisbot et al., 2007; Dragan et al., 2013).

Previous work related to sampling-based motion planning, particularly in the context of workspace-guided sampling-based planning and planning for high-DoF manipulators, is reviewed in the next section. Next, in Section 4, the workspace-guided sampling and connection primitives used by XXL are detailed, followed by a complete description of the planning algorithm in Section 5. The experimental evaluation of XXL appears in Section 6, and a discussion of sampling-based heuristics is presented in Section 7. This paper concludes with final remarks in Section 8.

2 Related Work

Sampling-based motion planning algorithms operate by constructing a discrete graph that approximates the connectivity of the robot's continuous *configuration space* (Choset et al., 2005; LaValle, 2006). Unlike classical planning techniques, sampling-based methods scale to higher dimensional planning problems by implicitly searching or probing the configuration space. The implicit search, however, leaves these algorithms at a distinct disadvantage since sampling-based approaches are at best *probabilistically complete* and cannot recognize a problem without a solution. At the heart of sampling-based algorithms are heuristics to *sample* a valid (e.g., collision-free) robot configuration and *connect* the sampled configuration to the graph via a valid motion. Most existing techniques derive such heuristics based on a distance or density metric defined over the robot's configuration space.

The distance between two robot configurations is arguably the most popular heuristic used in sampling-based algorithms due to its simplicity. The Probabilistic Roadmap Method (PRM) (Kavraki et al., 1996) attempts connections between a sampled configuration and the k-nearest existing configurations. The Rapidly-exploring Random Tree (RRT) and bi-directional RRT-CONNECT algorithms (LaValle and Kuffner, 2001) attempt a connection between a newly sampled configuration and the nearest configuration in the search tree. Defining a meaningful distance metric and quickly computing the neighborhood of a configuration is notoriously difficult, however, as the dimensionality of the search space becomes large (Weber et al., 1998; Aggarwal et al., 2001; Cheng and LaValle, 2001).

Another kind of sampling-based methods are those built upon the theory of expansive spaces that select a configuration for expansion with probability inversely proportional to the density of existing configurations in the local neighborhood (Hsu et al., 1999). Once a configuration is selected, a new configuration is sampled in the neighborhood of the selected configuration and a connection is attempted. Naïve density computations involve counting the number of configurations within a prescribed radius, but strong reliance on a distance measure suffers from the same issues discussed earlier with RRT-based methods. More sophisticated density estimates utilize a *projection* that maps an element of the configuration space to a lower dimensional Euclidean space where distance is well defined. SBL (Sánchez and Latombe, 2001) and KPIECE (Şucan and Kavraki, 2012), for example, estimate density by counting the number of configurations that project to the same cell in a low dimensional Euclidean grid. Care must be taken when defining the projection, however, so that the density estimate inferred is meaningful in the configuration space. For example, the default projection functions defined in the MoveIt! software for robot planning and execution (Şucan and Chitta, 2012) are tuned for serial manipulators, but empirically the same defaults do not have the same efficacy when planning for high-DoF humanoid systems. Works like STRIDE (Gipson et al., 2013) blur the distance/density divide. In STRIDE, the properties of a sophisticated nearest-neighbor data structure is exploited to estimate the density of sampled configurations in a search tree and bias exploration to the *frontier* of the search space for high-DoF kinematic chains.

One widely used approach to improve the efficacy of sampling-based methods feeds workspace in-

formation back into the sampling and connection strategies to accelerate the search. Using workspace information to guide the configuration-space search is especially useful when there exists narrow passages in the configuration space that must be navigated. One popular method biases configuration space samples to lie on the *medial axis* of the workspace to maximize clearance from objects. Methods to sample the medial axis include retraction techniques (Wilmarth et al., 1999) or by pre-computing a surface that approximates the medial axis and then planning on this surface (Holleman and Kavraki, 2000). More recent work presents a technique to generate samples uniformly along the medial axis by computing the intersection of a randomly sampled line segment from the configuration space with the medial axis (Yeh et al., 2014), departing from the retraction-based approaches. Another related approach uses the generalized Voronoi diagram for a point in the workspace to inform configuration space planning for a rigid body (Foskey et al., 2001).

Spatial partitioning and decomposition techniques have also been used in a variety of ways to improve sampling and connection strategies in sampling-based algorithms. In PDST (Ladd and Kavraki, 2005), a partition is actively refined where sampled configurations occupy a unique cell in a polygonal decomposition of the workspace, and a sampling density measure is implicitly encoded via the volume of the decomposition regions. Octree decompositions of the workspace have been shown to accelerate the motion planning process by identifying narrow passages in the workspace and biasing the sampling procedure to generate configurations that project into difficult areas (van den Berg and Overmars, 2005). A triangulation of the workspace has been used to bias sampling toward narrow passages (Kurniawati and Hsu, 2004) and subsequent work uses adjacency information in the triangulation to find connections between disjoint portions of the roadmap (Kurniawati and Hsu, 2008). A graph of implicit configuration space regions can also be defined over a set of valid and invalid samples (Rodriguez et al., 2006). In this formulation, sampling is biased toward narrow passages in the configuration space by identifying regions with a high entropy, defined as the ratio of invalid to valid configurations within the region. Connections are also attempted along a region path to further accelerate the planning process in this technique.

A workspace decomposition can also be employed to select entire subsets of the configuration space for sampling and expansion. A spherical decomposition of the workspace computed via greedy wavefront expansion has been used to inform motion planning via focused sampling into a specific sphere and connecting configurations that project along a path of overlapping spheres (Brock and Kavraki, 2001; Yang and Brock, 2005). SYCLOP (Plaku et al., 2010) and GUST (Plaku, 2015) establish a link between the robot's workspace and configuration space during the search to accelerate the motion planning process for mobile robots with differential constraints. In these methods, sampling and connection are restricted to a minimal-cost series of promising workspace regions until a solution path is found. In this context, the cost is a heuristic that attempts to capture the difficulty of traversing a region. The cost is dynamically updated for each edge of the workspace decomposition to inform the motion planner about features such as the number of samples in a region, the dispersion of the samples in the region, and the number of times a region has been selected for expansion.

The works described above focus primarily on planning for a free-flying rigid bodies or mobile robots with dimensionality significantly less than a mobile manipulator platform. Strategies using workspace information specially tailored for manipulator platforms have also been used to select desirable goal configurations (Bertram et al., 2006) or to bias the search in the general direction of the goal (Weghe et al., 2007; Diankov et al., 2008). Sampling techniques for manipulators that reparametrize the problem in terms of the workspace distance between links are effective for generating valid configurations for high-DoF manipulators, particularly those with closed-loops (Tang et al., 2010). Scalable planners for high-DoF manipulators that plan directly in the workspace are effective particularly when the workspace is relatively uncluttered.. In (Shkolnik and Tedrake, 2009), the authors' propose the *task-space* RRT (TSRRT) that performs nearest-neighbor checks after projecting a manipulator configuration into the workspace and demonstrate fast computation times for planar kinematic chains with

hundreds of joints. Such techniques can also be extended to mobile manipulator platforms by relying on the *Jacobian* to guide the search through the workspace (Rickert et al., 2014). The greediness of Jacobian-based methods, however, sacrifices any completeness properties for these algorithms. As the dimensionality of the manipulator grows, many (possibly infinitely many) inverse kinematics solutions may exist for the manipulator to reach a particular workspace position. In such cases, an early decision can guide the search to a local minimum from which a solution cannot be found.

A number of related works exist that augment sampling-based algorithms for instances of *whole-body* manipulation for humanoids (e.g., Kuffner Jr. et al., 2002; Hauser et al., 2008; Yoshida et al., 2010; Berenson et al., 2011; Burget et al., 2013; Hauser, 2014). The primary concern in these works is the generation of dynamically-stable motions for high-DoF bipedal robots, an effort that is complementary to the work presented here. The extension of this work to dynamically-stable systems is briefly discussed in Section 8.

Finally, although a great deal of effort has been put into improving both the efficiency and solution quality of sampling-based motion planners, these methods still have a reputation for computing poor quality paths due to the inherent randomness of the sampling process. In situations when the quality of a path is critical, like R2 navigating the ISS, sampling-based algorithms are often amenable to additional heuristics and refinement procedures to derive improved techniques that yield solutions of much higher quality. In TRRT (Jaillet et al., 2010; Devaurs et al., 2013), each sampled configuration is scored and configurations with cost larger than their parent are filtered probabilistically using the Metropolis-Hastings criterion (Metropolis et al., 1953; Hastings, 1970); the likelihood of a high-cost configuration being accepted in TRRT reduces as the search progresses. Sampling-based algorithms also exist that seek to minimize a cost metric defined over the entire path. PRM* and RRT* (Karaman and Frazzoli, 2011) are examples of *asymptotically optimal* algorithms that converge to a minimum-cost (e.g., shortest) path given infinite computing time. The sampling-based A* algorithm (Persson and Sharf, 2014), loosely based upon the notion of expansive spaces, is another example of an asymptotically optimal planner that converges to an optimal solution in the limit by quantifying the information gain of a sampled configuration. When both configuration and path quality are of import, TRRT* (Devaurs et al., 2014) provides a principled framework that asymptotically converges to the optimal solution, although effectively combining scalar cost (potential) functions is notoriously difficult in the context of motion planning (LaValle, 2006). While these techniques often return solutions with significantly better quality relative to methods that do not reason over a cost metric, the operations employed by the optimizing planners to discover such a solution can be expensive to compute. TRRT can be slow to discover a solution specifically due to the Metropolis-Hastings rejection criterion, and asymptotically optimal methods often converge slowly to a high-quality path due to a large number of uninformed invocations of the local connector, particularly in high-DoF planning problems (Luna et al., 2013).

3 Contributions

The proposed algorithm in this work, XXL, is probabilistically complete under mild technical assumptions and builds upon the prior works in decomposition-guided sampling-based motion planning for high-DoF manipulators in several ways. First, XXL mitigates the pitfalls of relying on a single projection function to estimate density in the configuration space by projecting a series of points on the manipulator into a decomposition of the workspace. This allows XXL to estimate coverage at a finer resolution for high-DoF manipulators by reasoning over the location of an end-effector, for example, when the position of a proximal link is contained within a specific decomposition region. In addition, XXL explicitly focuses the sampling and connection procedures into portions of the configuration space that project a specific point on the robot into a contiguous path of workspace regions, similar to (Kurniawati and Hsu, 2008; Plaku et al., 2010). Unlike prior works, the path of regions is not specific

to a single projection function; XXL selects one point on the robot to guide through the regions on each iteration. A partial ordering of the points naturally emerges from the kinematic hierarchy of the robot. The position of an end-effector, for example, depends on the location of all proximal links in the manipulator, and this hierarchy is traversed by XXL during the planning process.

Simulated planning experiments for R2 in the confined ISS module demonstrate that XXL quickly computes valid paths for 14-DoF (two legs) and 21-DoF (two legs and one arm) scenarios. Moreover, the novel sampling and connection strategies employed by XXL result in superior quality motions compared to many existing methods, largely avoiding the undesirable somersaults while discovering paths where the end-effector(s) travel significantly shorter distances through the workspace, all without explicitly optimizing over any path-quality metric. Evaluation of XXL in planar manipulator planning scenarios further illustrates the generality and scalability of the proposed planning algorithm. Finally, an extended discussion on the efficacy of two popular heuristics in sampling-based motion planning is conducted that illustrates some limitations of guiding heuristics in sampling-based algorithms for manipulators as the number of DoFs increases.

4 Workspace-guided Planning Primitives

Projecting whole robot configurations as points into a decomposed 2D/3D workspace has been shown in prior sampling-based algorithms to be an informative signal to assess search coverage and progress toward a goal configuration (e.g., Sánchez and Latombe, 2001; Ladd and Kavraki, 2005; Kurniawati and Hsu, 2004, 2008; Şucan and Kavraki, 2012). The number of configurations that project into a specific decomposition region is one measure of density, for example. The workspace decomposition can be as simple as a uniform grid or a complex subdivision of the free workspace area. The type of decomposition is domain dependent, but the size and adjacency of the regions should adequately capture how the robot moves through the workspace. As the difference in dimension between the robot and its workspace grows, however, ambiguity in the workspace projection can lead to poor coverage and distance estimates.

Concretely, consider the planar manipulator in Figure 2a. The 4-DoF manipulator operates in a 2D workspace, making the projection surjective (distinctness is not preserved) and introducing errors in the density computation. Presume that the workspace projection for the planar manipulator is the position of the end-effector. As shown in Figure 2b, projecting the end-effector position of a kinematically redundant manipulator leads to inaccurate density computations when distinct configurations project to the same point in the lower dimensional projection space. An assumption made in both this work and related algorithms that utilize a projection is that, with high likelihood, the relative distance between configurations is nearly preserved in the projection space. The well-known Johnson-Lindenstrauss Lemma (Johnson and Lindenstrauss, 1984) provides a theoretical basis for computing such a projection. Unfortunately, motion planning algorithms are not developed directly on this result due to the difficulty in computing the distance-preserving embedding, an area of active research in several different fields (e.g., Achlioptas (2001); Şucan and Kavraki (2009); Ailon and Chazelle (2009); Andoni et al. (2014)).

To mitigate errors in the density computation for high-DoF manipulators, a series of points on the system can be projected into the workspace, rather than a single projection, forming a higher-dimensional projection space from which a more informed density metric is derived. Observe from Figure 2c that projecting the position of an intermediate link is useful for disambiguating configurations where the end-effector projection is the same. Several existing works support a higher dimensional projection function in theory to assist in assessing the frontier of the search space at the expense of computation time (e.g., Sánchez and Latombe (2001); Şucan and Kavraki (2012)), but our work advocates an even tighter coupling between the workspace decomposition and the configuration space

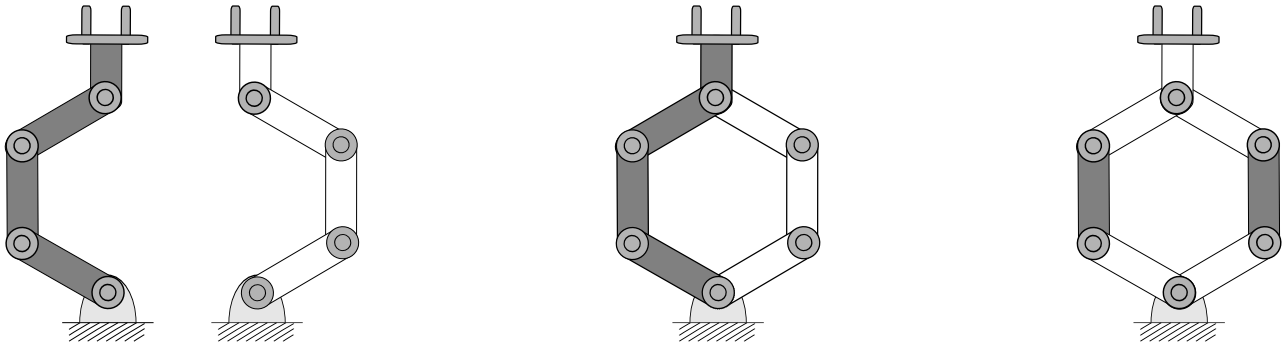


Figure 2: The end-effector position is an intuitive projection function for a robotic manipulator but can result in poor coverage estimates in the configuration space. (a) Two configurations for a planar kinematic chain with four joints are far apart in the configuration space, but have the same end-effector position, shown in (b). (c) Projecting a position on an intermediate link (shaded) disambiguates different configurations with the same end-effector position.

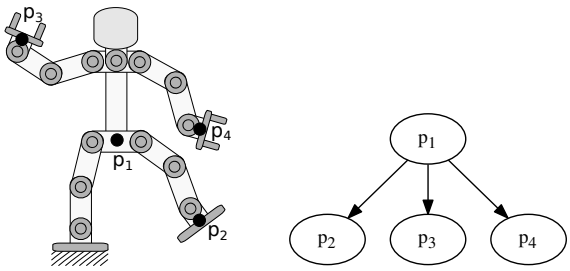
search. In this section, sampling and connection primitives are presented that make extensive use of multiple projection functions and the workspace decomposition to guide the configuration space search for a high-DoF manipulator.

4.1 Projection dependency graph

Given a set of points P on the geometry of the robot, positional dependencies among the elements of P are exploited by the workspace-guided sampling and connection procedures to accelerate the configuration space search. These dependencies are represented via the *projection dependency graph* (PDG) where an edge $p_i \rightarrow p_j$ indicates that the position of p_j depends on the position of p_i. Formally, it is assumed that the robot consists of a set of rigid links connected together with actuated joints, and that one link of the robot is always rigidly attached to the environment. The attached link is referred to as the *root* link. The connections between links form the *kinematic hierarchy* of the robot and can be modeled as a graph where the vertices consist of the links and an edge represents a joint connecting two links. Let $P = \{p_1, p_2, \ldots, p_n\}$ denote a set of points on the robot that are projected into the workspace where each point lies on a unique link of the robot. From the kinematic hierarchy, a partial ordering of P can be derived. Let l_i and l_j denote the links that contain the points p_i and p_j, respectively. The position of p_j depends on p_i if there exists a simple path in the kinematic hierarchy from the root link to l_j that passes through l_i. Using this definition, the PDG is constructed where P comprises the vertices, and a directed edge (p_i, p_j) exists iff p_j depends on p_i. If the robot does not have any closed chains, then by construction the PDG is a directed, acyclic graph.

An example PDG for a humanoid system is shown in Figure 3. The selection of points on the robot can easily be automated using a set of simple rules such as all end-effectors, the midpoint of long serial chains, links with multiple children, etc. A natural PDG often emerges from the kinematic reachability and redundancy in the system. The number of DoFs and the dimensionality of the workspace ($\mathcal{D}_w$) induces a practical upper-bound on the number of points $\lceil \frac{DoFs}{\mathcal{D}_w} \rceil$ for maximum reachability. Let d_p be the distance along the kinematic hierarchy from the root to point p in the PDG. Although not strictly required, ideally each point p can reach all positions within d_p of the root in the workspace.

Utilizing the PDG, informed sampling and connection strategies are next derived to focus the sampling procedure into specific subsets of the configuration space and intelligently connect disjoint portions of a roadmap.



(a) The set of points to project into workspace

(b) The PDG derived from the set of points

Figure 3: An example projection dependency graph given a predefined set of four points on the humanoid.

4.2 Workspace-guided sampling

With the PDG and a decomposition of the robot's workspace, an informed sampling strategy can be derived to explicitly sample portions of the configuration space that project specific points in the PDG into a particular regions of the workspace. For the remainder of this work it is assumed that the PDG is a directed, acyclic graph; refer to Section 8 for a discussion of extending this work to systems with closed chains. The key observation for the workspace-guided sampling primitive is that when the kinematic hierarchy of the system is acyclic, a path from the root to a leaf node in the PDG forms a kinematic chain that can be sampled using standard inverse kinematics methods.

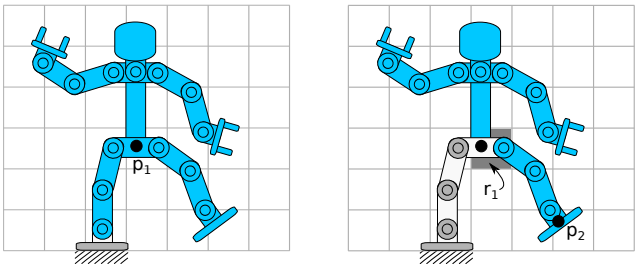
Formally, let $T = ((p_1, r_1), \ldots, (p_m, r_m))$ denote a topologically-ordered *traversal* of the PDG that annotates each point p_i with a specific workspace region r_i. Given T, the objective of the workspace-guided sampler is to generate a complete configuration of the robot such that point p_i projects to workspace region r_i. Note that T may only be a partial traversal of the PDG. In this case, points not specified in T are free to occupy any region of the workspace. Implementation of the workspace sampler can be performed without rejection using standard inverse kinematics (IK) techniques (Murray et al., 1994; Sciavicco and Siciliano, 1996; Wang and Chen, 1991; Aristidou and Lasenby, 2011).

Figure 4 illustrates the sampling procedure as the example PDG is traversed. The traversal is initially empty, meaning that all DoFs are unconstrained. Once p_1 is constrained to a specific region in the workspace, all DoFs from the root to p_1 are essentially immutable since changes in these joint angles can move p_1 to a different workspace region and violate the traversal. As subsequent points in the PDG are placed into particular regions, the total number of free DoFs dwindles until virtually all DoFs are constrained as the last point in the PDG is placed.

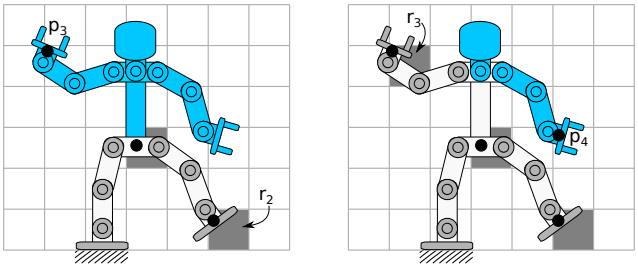
Note that the traversal is advantageous from a implementation point-of-view since most numerical IK solvers rely on a *seed* configuration to initialize the search. As the PDG is traversed, immutable DoFs may be omitted from IK so long as the seed configuration satisfies the traversal. This results in a smaller search space for the sampler as the system is constrained while trivially ensuring the sampler maintains the immutable-DoF invariant in the traversal. The implementation and evaluation of this work makes use of this assumption.

4.3 Workspace-guided connection

The workspace decomposition can also be used to intelligently connect existing configurations between workspace decomposition regions. In this work, sampling and connection are restricted to configurations that project a particular point in the PDG within a path connecting adjacent workspace regions, referred to as the *lead*. When computing a lead, a point in the PDG is selected given the current traver-



(a) $T = ()$ (b) $T = ((p_1, r_1))$



(c) $T = ((p_1, r_1), (p_2, r_2))$ (d) $T = ((p_1, r_1), (p_2, r_2), (p_3, r_3))$

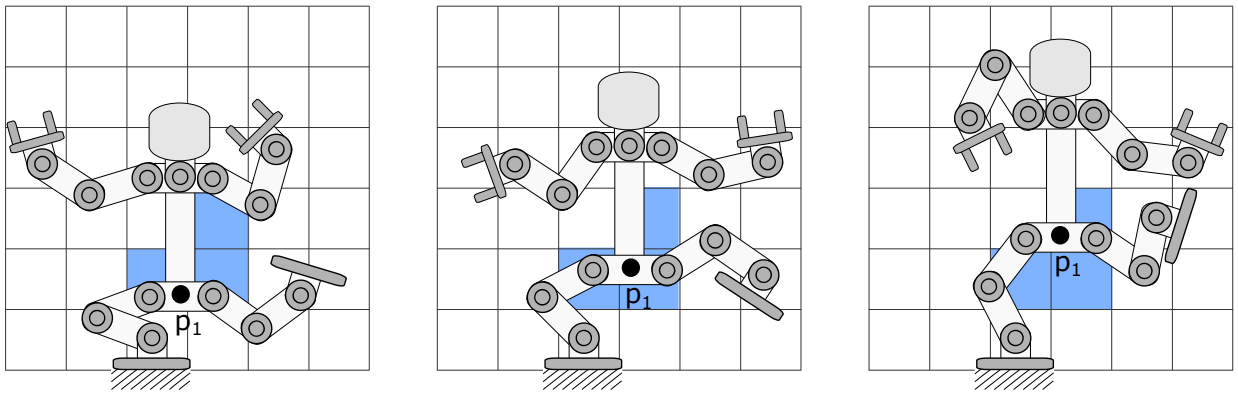
Figure 4: An illustration of the DoFs that may be sampled (shaded) at each node in the PDG from Figure 3 given the topological ordering (p_1, p_2, p_3, p_4) and grid decomposition of the workspace. All DoFs may be sampled when planning for the first point in a PDG since $T = ()$. Although the topological ordering is not unique, other orderings result in a similar segmentation of the DoFs.

sal T such that the PDG is traversed in topological order. Specific details of the lead computation are presented in Section 5.2.

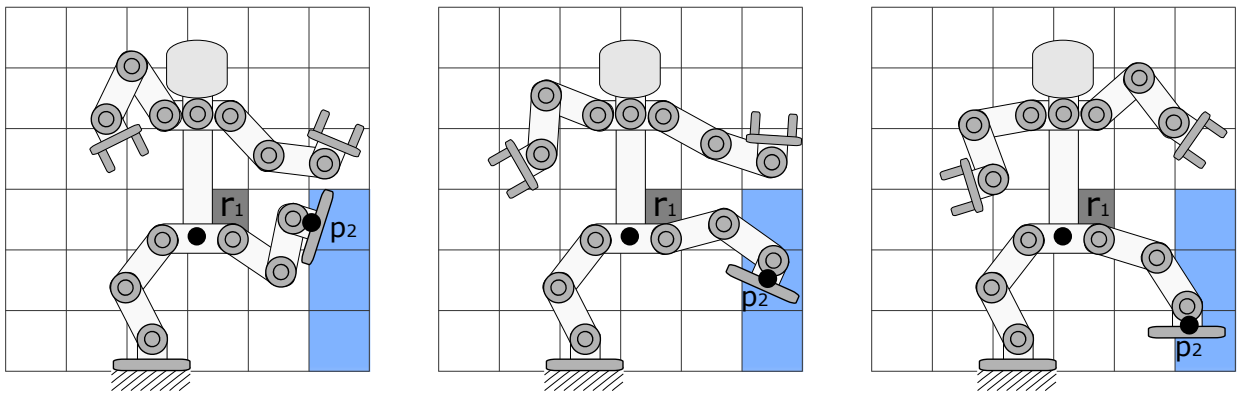
Connecting configurations along a lead is best explained through an illustration. Consider Figure 5, which demonstrates a two-node traversal of the PDG from Figure 3. In Figure 5a, p_1 is chosen for planning since it is topologically the first point in the PDG. A discrete lead is computed from the initial condition to a goal region, indicated by the shaded blue regions. Complete configurations of the robot are then sampled that project p_1 to the regions along the lead. Once there exists at least one valid configuration that places p_1 in every region of the lead, local connections are attempted between configurations in adjacent regions. If a valid path is discovered that brings p_1 to the goal region along the lead, the connection process is successful and sampling and connections may continue for the next point in the PDG. Figure 5b shows how this process might continue for p_2 while keeping p_1 locked to r_1.

5 XXL: Motion Planning for High-DoF Kinematic Systems

XXL is a probabilistically complete sampling-based motion planning algorithm specially tailored for high-DoF manipulator platforms. At its core, XXL constructs a roadmap that approximates the connectivity of the robot's valid configuration space by repeatedly traversing the PDG, computing a new lead for the selected point in the PDG, then sampling and connecting configurations along the lead using the primitives described in the preceding section to incrementally move a high-DoF manipulator toward a goal configuration. XXL maintains a set of weights over the decomposition for each point in the PDG, updating the weights on-the-fly to inform the lead computation about regions that are well explored, have been difficult to explore, and unexplored. A complete presentation of XXL is given in this section, followed by a proof of probabilistic completeness.



(a) $T = ()$; Planning for p_1.



(b) $T = ((p_1, r_1))$; Planning for p_2 while keeping p_1 in r_1.

Figure 5: Example connections along a lead for a partial traversal of the PDG in Figure 3. (a) Sampling and connection for p_1 on the humanoid's waist. The lead is the blue shaded regions containing p_1. The traversal is empty to indicating there are no prior constraints on any other points in the PDG. (b) Sampling and connection for p_2 on the robot's right foot. The non-empty traversal indicates that p_1 must remain in r_1.

5.1 Motion planning with XXL

The key operations of XXL are presented in Algorithm 1. XXL maintains a topologically-ordered traversal T of the PDG $\mathcal{P}$, and selects the next point p to plan for via the $\mathcal{P}$.`First` and $\mathcal{P}$.`Next` methods. In each iteration, a minimum-weight lead is computed for p (`ComputeLead`, line 6) that brings the selected point from a region connected to the initial configuration to a region connected to a goal configuration. Given a lead, valid configurations for the robot are sampled using the workspace-guided sampling scheme that project p into regions along the lead while maintaining any constraints encoded in T from prior iterations (`SampleRegions`, line 7). Connections are then performed among the configurations that project p into regions along the lead (`ConnectRegions`, line 8); a typical *local connector* from the sampling-based literature is sufficient. Finally, progress (or lack of progress) along the lead is encoded by updating weights in the decomposition to inform future lead computations (`UpdateWeights`, line 9). Details of the lead computation and weight update scheme are presented in the next section.

At the end of each iteration, XXL checks to see if a solution path exists in the roadmap (`SolutionPath`, line 10). If no solution exists, p is encoded in T along with the region r furthest along the lead that p was able to reach, and XXL repeats this process in a subsequent iteration for the next point in topological order in the PDG (lines 11-19). Algorithm 1 utilizes `LeadIsConnected` which returns true when connections exist between all regions along the lead (line 11), as well as `DisconnectedRegions` (line 17) which returns true when two consecutive regions along the lead are not connected. If p is the final point in the PDG to plan for, the traversal is cleared and XXL resumes the search with an empty traversal (lines 20-22).

Algorithm 1: XXL

Input: $\mathcal{P}$: PDG; $\mathcal{D}$: Workspace decomposition; q_{init}, Initial state; Q_{goal}: Set of goal configurations

```
1  (V, E) ← (q_init, ∅)                                          // initialize roadmap
2  π ← ∅                                                         // initialize solution path
3  T ← ∅                                                         // initialize traversal
4  p ← P.First()                                                 // select topologically first point from P
5  while π = ∅ do
6      l ← ComputeLead(T, D, p)                                  // compute a lead for p to reach goal
7      V ← V ∪ SampleRegions(T, l, p)                            // sample configurations for p along l that satisfy T
8      E ← E ∪ ConnectRegions(T, l, p)                           // connect configurations for p along l that satisfy T
9      UpdateWeights(T, D, l)                                    // update weights for affected regions along l
10     π ← SolutionPath(V, E, q_init, Q_goal)                    // check for a solution path
11     if LeadIsConnected(l) then                                // update the traversal: fully-connected lead
12         T.Append((p, l.End()))                                // constrain p to the final region along l
13         p ← P.Next(T)                                         // select topologically next point from P given T
14     else                                                      // update the traversal: partially-connected lead
15         for r ∈ l do                                          // iterate through the regions of l in order
16             r' ← l.Next(r)
17             if DisconnectedRegions(r, r', p, T) then          // Regions r and r' are disjoint wrt p
18                 T.Append((p, r))                              // constrain p to the last connected region along l
19                 p ← P.Next(T)                                 // select topologically next point from P given T
20     if T completely traverses P then                          // restart PDG traversal if all nodes explored
21         T ← ∅
22         p ← P.First()                                         // select topologically first point from P
23 return π
```

5.2 Lead computation

Computing a high quality lead in the workspace decomposition is key to effective roadmap construction in XXL. To assist in the lead computation, exploration information is shared between the roadmap in the configuration space and the workspace decomposition in the form of weights assigned to each workspace region. Since the same decomposition is used to compute a lead for every point defined in the PDG, the weight of each region depends on the traversal T and the point being planned for since a specific region may or may not be reachable by a point on the system given the spatial constraints imposed on other points in T.

Formally, a weight $w_T^p(r) \in \mathbb{R}$ is assigned to region r when planning for point $p \in$ PDG given the traversal T. This work advocates a simple weighting scheme based on the number of vertices, edges, and times a region is selected for the lead. Weights may be chosen in other ways to achieve the desired configuration space exploration versus exploitation for the given application. Vertices and edges in the roadmap encode exploration progress, but in cases where expansion into a particular region proves difficult, the number of lead appearances disambiguates regions that have been searched extensively from those that have not. Given a traversal $T = ((p_1, r_1), \ldots, (p_m, r_m))$, let $V_T \subseteq V$ denote the configurations where point p_k projects to region $r_k, \forall k = 1, \ldots, m$, and $V_T^p(r)$ be the subset of V_T where a point $p \notin T$ projects to region r. Analogously, denote E_T and $E_T^p(r)$ as the subset of edges connected to at least one element of V_T and $V_T^p(r)$, respectively. Finally, let L_T^p be the number of leads computed for point p given traversal T, and $L_T^p(r)$ be the number of times region r appears in a lead given T. The weight $w_T^p(r)$ is computed as

$$w_T^p(r) = \mathrm{e}^{\alpha v} \times \mathrm{e}^{\beta c} \times (1.0 - \mathrm{e}^{\gamma l}), \tag{1}$$

where

$$v = \frac{\left|V_T^p(r)\right|}{|V_T|}, \qquad c = \frac{\left|E_T^p(r)\right|}{|E_T|}, \qquad l = \frac{L_T^p(r)}{L_T}, \tag{2}$$

and $\alpha, \beta, \gamma \in \mathbb{R}^-$ are constants that can be tuned to adjust exploration into new areas versus exploitation of areas already searched. The evaluation of this work uses the values $\alpha = -1$, $\beta = -10$, and $\gamma = -1$ to penalize leads that follow regions that are already well connected.

Weights are updated in XXL after sampling and connecting along each lead. Although a lead is computed for a specific point in the PDG, weights over the decomposition are a function of the PDG traversal and roadmap configurations. Since whole robot configurations are sampled by XXL, planning the motions of one point in the PDG can affect the future leads of all other points as well. Nevertheless, updating the weights can be performed efficiently since additions to the roadmap are restricted to configurations that project to the regions along the lead. To mitigate large changes in the weight landscape after a single iteration of XXL, weights are *nudged* in the direction indicated by (1) but are not necessarily set to the absolute value. Concretely, given the existing weight $w_T^p(r)$ and a new weight $w_T'^p(r)$ computed via (1), the new weight for the region is updated as

$$w_T^p(r) = w_T^p(r) + \eta(w_T'^p(r) - w_T^p(r)), \tag{3}$$

where $0 < \eta \leq 1$ is the learning rate; $\eta = 0.1$ is used in this work. In the evaluation of this work, the weights are also bounded to the interval $[0, 1] \in \mathbb{R}$ so that an admissible heuristic can be easily defined over the decomposition to improve the lead computation time.

Given the set of weights defined over the decomposition, a discrete lead through the workspace for a point $p \in$ PDG and traversal T is computed as a minimum-weight path starting from any region r where $|V_T^p(r)| > 0$ and ending at a region that contains a projection of p from Q_{goal}. When $T = \emptyset$, the lead starts from the region containing the projection of p from the initial state q_{init} to promote connectivity of the roadmap near the start configuration. To guarantee probabilistic completeness, XXL utilizes a random walk through the workspace decomposition starting at any region instead of a minimum weight lead with a small probability $\delta > 0$. In the evaluation of this work, $\delta = 0.05$.

5.3 Workspace-guided primitives

Workspace-guided sampling (`SampleRegions`) can be implemented using a pseudoinverse-based iterative IK solver (Sciavicco and Siciliano, 1996). When sampling along a lead, it is often advantageous to avoid generating configurations in a region that is already densely sampled. This often occurs near the start and goal configurations since these workspace regions are selected frequently for the lead. Using the same notation as above, oversampling of a particular region r is reduced by sampling k new configurations from r with probability

$$1.0 - \frac{\left|V_T^p(r)\right|}{|V_T|}.$$

The value $k = 10$ is used in the evaluation of this work

Similar to oversampling, attempting connections within and between regions that are already relatively well connected is unlikely to reveal a solution path. Moreover, because connections within and between regions are attempted repeatedly, a naïve implementation of `ConnectRegions` that checks all pairs of configurations wastes computation time since an invalid edge may be checked many times over the course of planning. Two straightforward additions are employed by this evaluation to avoid duplicating the computation in XXL. First, validating an edge between two configurations should only be performed once, and knowledge of an invalid edge can and should be stored to prevent duplicating this computation. Second, validating the motions between all pairs of configurations between two regions can still result in a lengthy computation, particularly if the workspace regions are large. In this evaluation, XXL attempts connections between two regions probabilistically based on the ratio of unconnected configurations within each region. Let $V_T^p(r)$ denote the set of configurations for a point p in the traversal T that project to region r, and $\bar{V}_T^p(r) \subseteq V_T^p(r)$ be the subset of configurations without any edges in the graph. Then, the probability $p_c(r_i, r_j)$ that XXL attempts connections between region r_i and r_j is

$$p_c(r_i, r_j) = \max_{k \in \{i,j\}} \frac{|\bar{V}_T^p(r_k)|}{|V_T^p(r_k)|}, \tag{4}$$

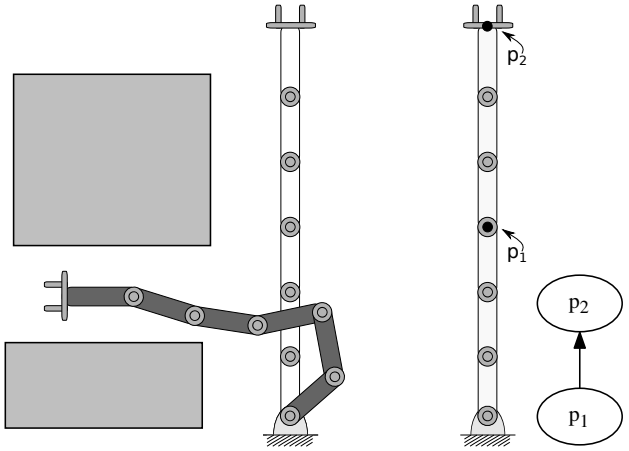
meaning that connections are more likely to be attempted when one of the regions has a large number of unconnected configurations.

5.4 Illustration

The main steps of Algorithm 1 are now demonstrated. Consider a six link planar manipulator with revolute joints as shown in Figure 6. The goal is to navigate the end-effector through the open space and into the corridor, as shown in Figure 6a. A two-point PDG containing the midpoint and end-effector, shown in Figure 7a, is used for illustration. Finally, a 2D grid decomposition, shown in Figure 6b, is used for projection and lead computation.

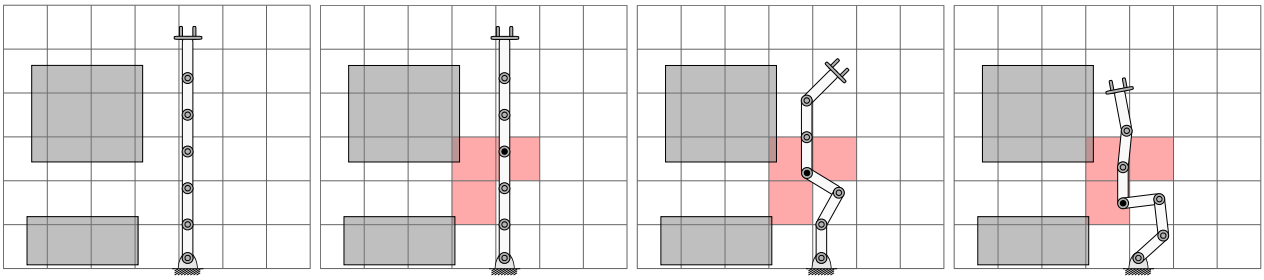
XXL begins by selecting the midpoint for planning since this point is topologically first in the PDG. A lead is computed to bring the midpoint from the region containing the initial state to one containing a projection of the goal state, as shown in Figure 7b. Workspace-guided sampling is invoked to generate valid configurations of the manipulator that project the midpoint into any of the regions along the lead. Connections between configurations in these regions are then attempted, and valid connections are encoded in the roadmap as edges. If successful, a partial solution path can be found that navigates the midpoint to the goal region, as shown in Figures 7c-7d.

After planning for the midpoint, XXL continues its traversal of the PDG and selects the end-effector point while fixing the midpoint to the workspace region containing a goal configuration identified in the previous iteration. XXL computes a lead for the end-effector, then performs the same sampling and connection procedures, taking care to keep the midpoint within the desired region. An illustration of



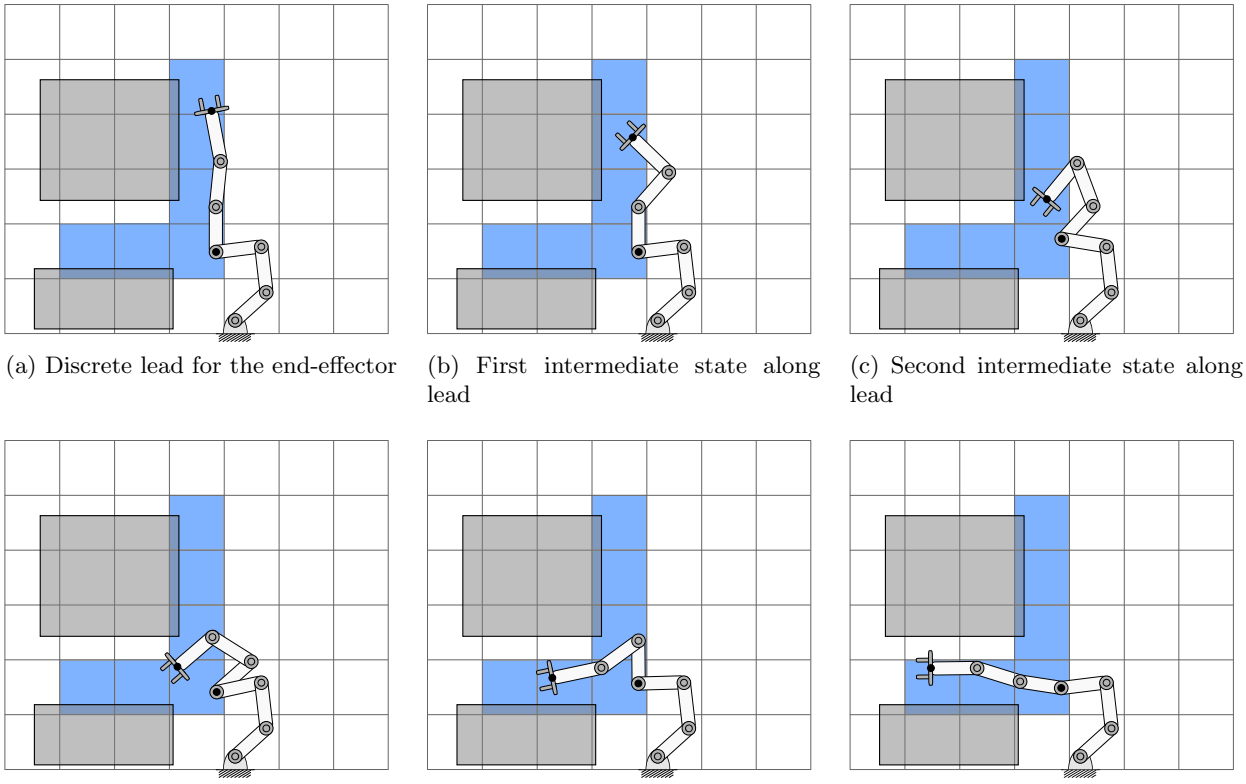
(a) Start and goal configurations (b) 2-point PDG

Figure 6: (a) The initial configuration and goal configuration (shaded) for a six joint planar manipulator in a workspace with two obstacles. (b) The midpoint and end-effector comprise the PDG in this example.



(a) Workspace grid decomposition $\mathcal{D}$ (b) Discrete lead for the midpoint of the manipulator (c) Intermediate state along lead (d) Final state along lead

Figure 7: One iteration of XXL for the manipulator in Figure 6a. A discrete lead through the workspace decomposition is computed (b) to bring the midpoint point from the region containing the initial configuration to the region containing the goal configuration (c-d).



(a) Discrete lead for the end-effector

(b) First intermediate state along lead

(c) Second intermediate state along lead

(d) Third intermediate state along lead

(e) Fourth intermediate state along lead

(f) Final state along lead

Figure 8: A subsequent iteration of XXL for the end-effector of the manipulator in Figure 6. A workspace lead (a) is computed to bring the end-effector from the initial region to the final region (b-f). The midpoint of the manipulator is not allowed to leave its current region.

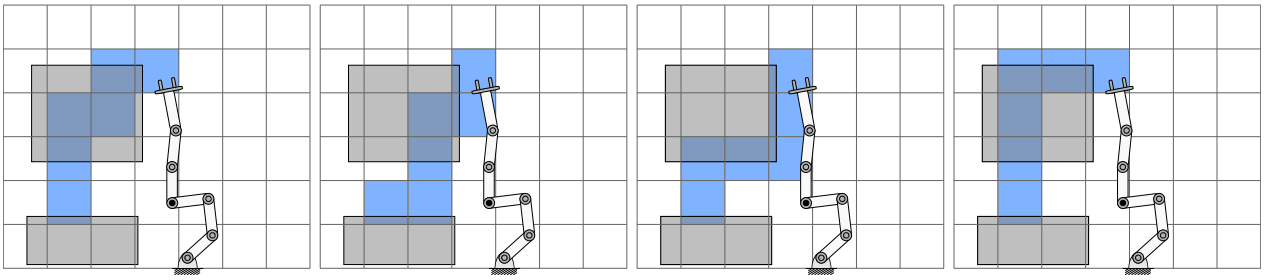


Figure 9: During the search, XXL may choose to explore leads that are difficult or impossible for the system to navigate since this information is not known *a priori*. Challenging parts of the space are encoded by increasing the weight of a region when sampling or connections show to be difficult, and future leads avoid high-weight regions when possible.

this subsequent iteration is shown in Figure 8. If a valid path for the end-effector is discovered at the end of sampling and connection, the complete solution path is returned by XXL.

Note that XXL is not expected to find the solution path in one PDG traversal. In particularly cluttered scenes, cases where the weights are not initialized to informed values, and/or when sampling is simply unlucky, XXL may need to traverse the PDG many times and try many different leads before a solution path is found. If the weights are uniformly initialized, XXL may need to search over many different leads before discovering those that are more likely to lead to a solution path. Figure 9 shows some possible leads that XXL may explore before discovering one that contains a solution path. There are direct improvements to improve the lead computation from this observation, including discarding regions that lie entirely within an obstacle and/or estimating the likelihood of generating valid samples in each region through precomputation. Further note that, by construction, XXL must sample and connect configurations along each region of a lead before a solution path can be found. A dense workspace decomposition (e.g., small cell sizes) is advantageous in cluttered scenes where the lead is informative in how to navigate through the environment. A similarly dense decomposition in an otherwise empty scene potentially disadvantages XXL relative to other sampling-based methods since many configurations must be explicitly sampled and connected even if there exists a simple solution path.

5.5 Probabilistic completeness

An algorithm is *probabilistically complete* if the probability of discovering a solution converges to 1 with increasing runtime, provided a solution exists. In this section, XXL is proven probabilistically complete when the workspace-guided sampler is able to generate all configurations of the robot. For simplicity, this analysis assumes a PDG with just one node, but completeness results trivially extend to a PDG with multiple nodes since, as shown in Figure 4, all DoFs may be sampled and connected in XXL when planning for the first point in the PDG. The analysis also assumes that connections between configurations follow straight paths through the configuration space. Probabilistic completeness is demonstrated by first showing that if the workspace-guided sampler is able to generate all configurations of the robot, XXL eventually samples the entire configuration space. The main result, Theorem 6, then shows XXL eventually connects the configurations along a solution path.

To begin the analysis, notation used throughout the proof is presented. Let Q denote the configuration space of the robot, $Q_{free} \subseteq Q$ as the collision-free subset of Q, and $\mathcal{W}$ as the Euclidean workspace of the robot. Define $\texttt{Proj} : Q \to \mathcal{W}$ as a projection that maps a robot configuration to a point in the workspace. Denote $Q_p = \{q \in Q \mid \texttt{Proj}(q) = p\}$ as the set of all configurations that project to a specific point in the workspace. The cardinality of Q_p is often greater than one when the manipulator

is kinematically redundant. Furthermore, let $\mathcal{S}$ be a workspace-guided sampling algorithm that takes as input a position $p \in \mathcal{W}$ and returns a configuration $q \in Q_p$ with probability $P_{\mathcal{S}}(q \mid p)$. It is assumed that configurations returned by $\mathcal{S}$ are drawn independently.

For XXL to be probabilistically complete, the sampler $\mathcal{S}$ must have a non-zero probability of returning any configuration $q \in Q_p$ given p. This property is formally stated in the following definition.

Definition 1. *A workspace-guided sampler $\mathcal{S}$ is redundancy-robust if, $\forall p \in \mathcal{W}, \forall q \in Q_p, P_{\mathcal{S}}(q \mid p) > 0$.*

Presuming the workspace-guided sampler employed by XXL is redundancy-robust, it is demonstrated that XXL samples from each region infinitely often. Using this result, it is then shown that XXL eventually samples the entire configuration space.

Lemma 2. XXL *selects each region in the workspace decomposition infinitely often for the lead until a solution path is found.*

Proof. This statement is proven by contradiction. Let R denote the set of workspace regions and $r \in R$ be a region that appears in the lead a finite number of times regardless of how many iterations XXL has run. Then at some iteration $i \in \mathbb{N}$ there does not exist a $c \in \mathbb{N}^+$ such that during iteration $i + c$, region r appears in the lead. Recall that XXL computes the lead via a random walk starting at an arbitrary region with a small probability $\delta > 0$. Then the probability that any region appears in the lead is at least $\frac{\delta}{|R|}$ during each iteration of XXL. This implies $\forall r \in R$, the expected value of c is finite and bounded by $\frac{|R|}{\delta}$, contradicting the original assumption. If c does not exist, then XXL terminates with a solution path. □

Under the assumptions that $\mathcal{S}$ is redundancy-robust and sampled configurations are drawn independently, the probability that $\mathcal{S}$ generates a specific configuration $q \in Q_p$ given p increases exponentially with the number of attempts. The exact probability is given in the following definition.

Definition 3. *For a redundancy-robust sampler $\mathcal{S}$ and input position p, the probability that $\mathcal{S}$ returns configuration $q \in Q_p$ after n attempts is*

$$1 - (1 - P_{\mathcal{S}}(q \mid p))^n. \tag{5}$$

Lemma 4. XXL *samples the entire configuration space when the workspace-guided sampler $\mathcal{S}$ is redundancy-robust unless a solution path is discovered.*

Proof. This statement follows from Lemma 2 and Equation 5. Since each region is selected infinitely often for the lead (Lemma 2) and XXL draws points from the lead uniformly as input to $\mathcal{S}$, it follows that all points in the workspace are also selected infinitely often as input to $\mathcal{S}$. Care must be taken to select points that lie exactly on the workspace region boundaries as well as those in the interior of each region as input to $\mathcal{S}$ to ensure coverage of the configuration space. Since the the probability of $\mathcal{S}$ returning a specific configuration under the same input increases exponentially with the number of attempts (Equation 5), and all inputs are selected infinitely often by XXL, the claim holds. □

Although the preceding lemma proves that XXL eventually samples the entire configuration space, what remains to show is that XXL eventually finds a solution path if one exists. The main result builds upon the probabilistic completeness properties of the PRM (Kavraki et al., 1998).

Definition 5. *(Karaman and Frazzoli, 2011): A motion planning problem is robustly feasible if there exists a positive constant ϵ and solution path $\sigma : [0, T] \to Q_{free}$ where $\forall 0 \leq t \leq T$, $\sigma(t)$ has a minimum distance of ϵ from the nearest obstacle.*

Theorem 6. XXL *eventually finds a solution to a robustly feasible motion planning problem when the workspace-guided sampler is redundancy-robust.*

Proof. Let $\sigma : [0, L] \rightarrow Q_{free}$ denote a valid, continuous solution path of length L for a robustly feasible planning problem where $\sigma(0)$ is the initial configuration and $\sigma(L)$ is a goal configuration of the robot. Denote $\sigma_\epsilon \subseteq Q_{free}$ as the set of configurations no further than ϵ from any configuration in σ. Without loss of generality, let $r_\sigma^1, \ldots, r_\sigma^n$ denote the sequence of workspace regions the robot passes through when executing σ, and $\sigma_\epsilon^i = \{q \in \sigma_\epsilon | \texttt{Reg}(q) = i\}$, where $\texttt{Reg}(q)$ returns the region ID that configuration q projects to in the workspace. Regions r_σ^i and r_σ^{i+1} must be adjacent since the robot moves rigidly and continuously through its workspace, implying that σ_ϵ^i and σ_ϵ^{i+1} are also adjacent in the configuration space.

Let $k = \max(n, \lceil \frac{2L}{\epsilon} \rceil)$. Further, denote $d(a, b)$ as the distance between $\sigma(a)$ and $\sigma(b)$ along σ. There exists a discrete set of k configurations $x_1 = q_{init}, x_2, \ldots, x_k = q_{goal}$ along σ such that $d(x_i, x_j) \leq \frac{\epsilon}{2}$ and where at least one x_i projects into each r_σ^j. Let $B_r(a)$ represent the set of configurations within a ball of radius r centered at $\sigma(a)$. For configurations x_i, x_{i+1} along σ, the straight-line motion between $q_i \in B_{\epsilon/2}(x_i)$ and $q_{i+1} \in B_{\epsilon/2}(x_{i+1})$ must also be valid since $B_{\epsilon/2}(x_i) \subset B_\epsilon(x_i)$ and $B_{\epsilon/2}(x_{i+1}) \subset B_\epsilon(x_i)$ (Theorem 1 from Kavraki et al. (1998)). Using this result, it is sufficient for XXL to sample at least one element of $B_{\epsilon/2}(x_i), \forall i \in \{1, 2, \ldots, k\}$ and connect these configurations via straight paths through the configuration space.

Presume that configurations $q_i \in B_{\epsilon/2}(x_i)$ and $q_{i+1} \in B_{\epsilon/2}(x_{i+1})$ have already been sampled by XXL. This eventually happens if a solution path exists, as shown in Lemma 4. By construction, q_i and q_{i+1} must project to adjacent regions r_σ^j and r_σ^{j+1} or project to the same region r_σ^j.

XXL attempts connections among configurations within a region and between adjacent regions immediately after sampling configurations along the lead (Algorithm 1, line 8). Since every region appears infinitely often in the lead (Lemma 2), it follows directly that all pairs of adjacent regions also appear infinitely often in the lead, and connections between all configurations that project to regions in the lead are eventually attempted by XXL. Therefore, the motion between q_i and q_{i+1} must eventually be found by XXL when attempting to connect configurations between regions r_σ^j and r_σ^{j+1} (or within r_σ^j).

Since every region is selected infinitely often for the lead, all configurations that project into the regions of the lead have a non-zero probability of being sampled, and a connection between all pairs of configurations within each region and between adjacent regions of the lead are eventually attempted, XXL will eventually discover a solution path when one exists. □

Although Theorem 6 shows that XXL is able to eventually find a solution when one exists, like most other sampling-based methods there are no strict convergence bounds. It is possible, however, to derive a worst-case bound in terms of the number of sampled configurations before a solution path is found (Kavraki et al., 1998; Ladd and Kavraki, 2004).

A redundancy-robust sampler as defined in Definition 1 is a strong technical assumption. Note, however, that under this assumption that the proof of Theorem 6 holds under any workspace decomposition, PDG, and cost function, implying that XXL is somewhat robust to sub-optimal inputs.

6 Evaluation

In this section, XXL is evaluated in a series of simulated planning experiments. All computations are performed on an 8-core, 4.0GHz Intel i7-6700K with 32GB of RAM. The implementation of XXL and all other planners is in C++ and uses the Open Motion Planning Library (Şucan et al., 2012). Simulation of R2 uses ROS Indigo (Quigley et al., 2009) and MoveIt! (Şucan and Chitta, 2012).

Throughout this evaluation, planning algorithms are evaluated for not only the speed at which a solution is found, but also the quality of the solution path. Two different criteria are utilized to estimate the quality of a solution path. The Cartesian distance traveled by a set of points on the robot through the workspace provides a whole-path metric that is ideally minimized. A configuration-level metric is also employed to identify undesirable torso orientations for R2 like those in Figure 1b. These configurations are identified by inspecting the *declination* angle of R2's torso from vertical relative to a fixed coordinate frame. Let θ_d denote the declination of R2 at some configuration. The declination is defined in the range $0 \leq \theta_d \leq \pi$, where a value of 0 indicates a perfectly vertical torso, and a value of π indicates the torso has tumbled vertically over itself. The maximum declination angle along the solution path for R2 is ideally minimized to ensure an upright R2.

First, a sequence of steps are planned for the R2 platform inside the Destiny module of the International Space Station using a gradient descent method to illustrate the difficulty of the competing computation time and path quality metrics, followed by an evaluation of XXL in this setting compared to a broad set of traditional sampling-based motion planning algorithms. XXL is then compared with TRRT (Jaillet et al., 2010; Devaurs et al., 2013) and RRT* (Karaman and Frazzoli, 2011) for R2 performing the same steps. These methods are specifically chosen since they are able to reason over the declination and Cartesian distance measures respectively to potentially obtain higher quality solutions. The robustness of XXL to parameters in the workspace decomposition and lead computation is then examined when planning for R2. Finally, XXL is evaluated on a planar kinematic chain system in two different environments with varying degrees of freedom to demonstrate the scalability of the algorithm to systems with dozens of DoFs.

6.1 Planning for Robonaut2

In this section, a series of steps is planned for R2 in a module of the International Space Station beginning at its *unstow* pose as it exits the storage rack at one end of the module. R2 is restricted to grasping a set of handrails attached to the surfaces of the module with a specialized gripper on each of its feet, and one foot must be rigidly attached at all times. For simplicity of evaluation, each of the steps are specified independent of one another. Each step has a predefined initial configuration and a desired goal pose for the unattached foot link. Furthermore, R2's torso and head must be vertical (zero roll and pitch) at the end of each step.

To illustrate scalability of XXL, planning experiments for R2's legs and R2's legs and one arm are performed, corresponding to 14 and 21 DoFs respectively. Figure 10 shows the three steps for the 14-DoF evaluation. In these simulations, step 1 is given a timeout of 10 seconds, and the two subsequent steps have a timeout of 30 seconds since the required motions are considerably more complex. R2 must maneuver its right leg out of the storage rack in the second step to grasp a handrail on the wall, and in the third step must navigate a configuration space riddled with self-intersections as both legs are moving while avoiding collisions with the module.

Simulated steps for the 21-DoF configurations begin at the same unstow pose as in the 14-DoF case, but require the system to avoid a large region near the unstow pose and also move the right hand to a particular pose. Similar to the 14-DoF experiments, the steps are specified independent of one another with a predefined initial configuration and a goal pose for the foot and hand links; R2's torso must also be vertical at the end of each step. Due to the increased dimensionality of the search space and complexity of the scene, a planning timeout of 30 seconds is used for the first step and 60 seconds for the second step. An illustration of these steps is shown in Figure 11.

It is important to emphasize that R2 is relatively cramped within the ISS. Although the upper body of R2 is humanoid in size, its legs are not designed for terrestrial navigation and are about 1.4m long for maximum reachability on the ISS. In addition, R2's arms are about 0.8m long and its torso has a height of roughly 1m (waist to head) with a shoulder width of about 0.8m. For comparison,

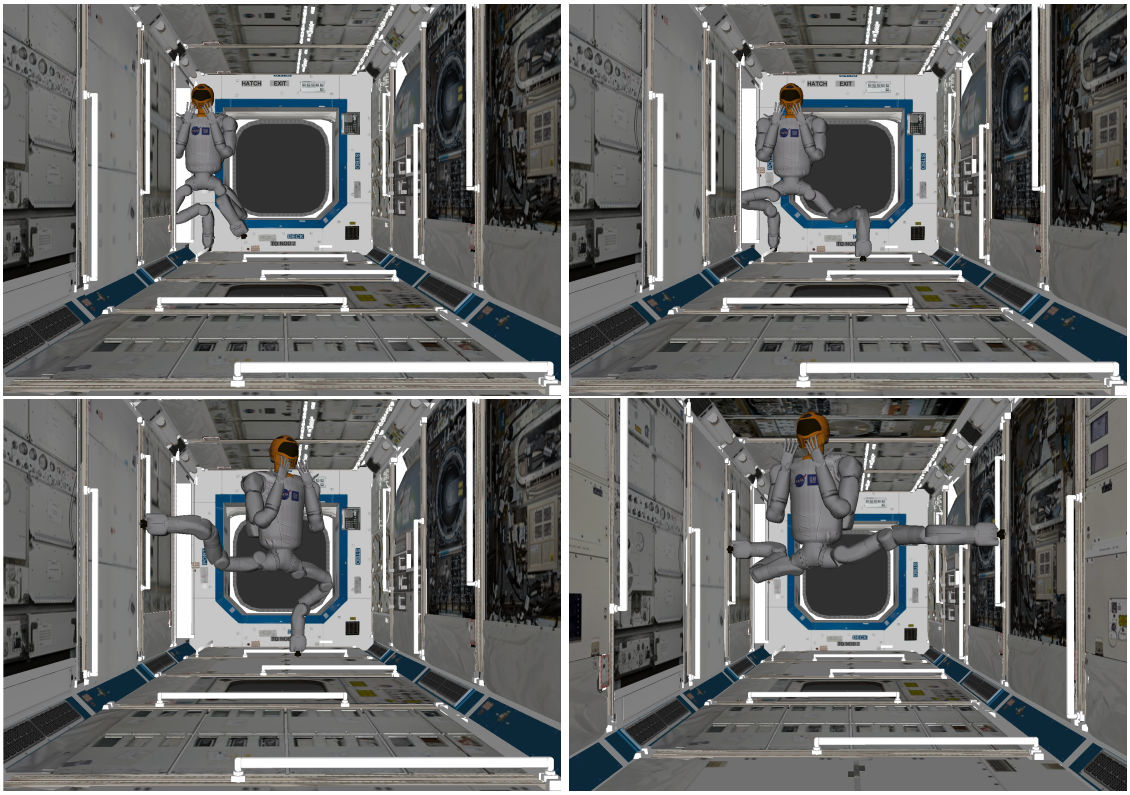


Figure 10: The *unstow* pose of R2 in the Destiny module (top left), followed by the three sequential steps R2 must take in the ISS (14 DoFs). Handrails that are used by the robot (and astronauts) are shown in white.

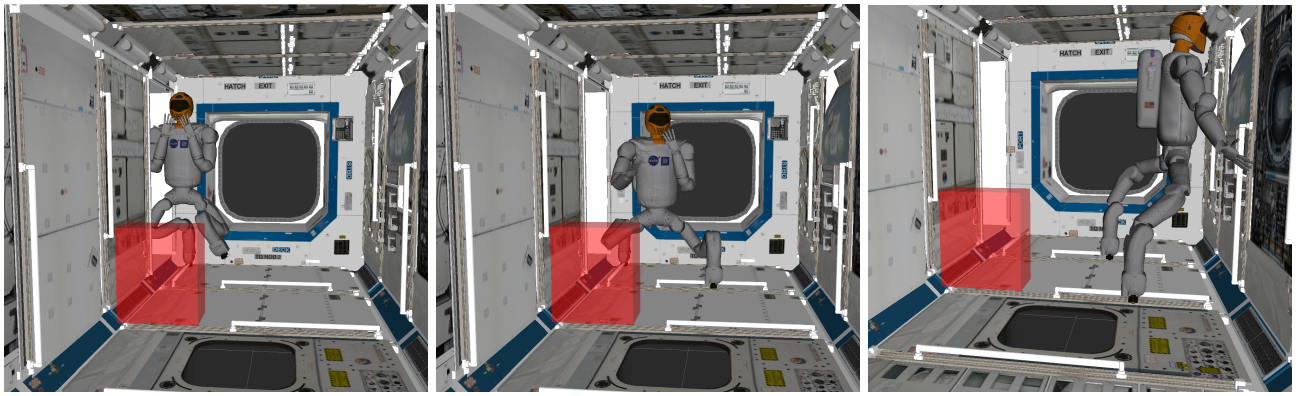


Figure 11: The initial *unstow* pose of R2 in the Destiny module, followed by the two steps R2 must take in the ISS while moving the right arm (21 total DoFs) and avoiding the rectangular obstacle.

the Destiny module has an interior width and height of only about 2.3m after accounting for storage racks.

6.1.1 Gradient descent

In this first experiment, a randomized gradient descent technique is employed to locally perturb an initial, possibly invalid path for R2 that connects start to goal in order to arrive at a valid solution path. The objective is to illustrate the tradeoffs when optimizing over competing criteria (e.g., computation time and path quality) in this planning domain. The gradient descent method, detailed in Algorithm 2, parameterizes the path as a series of discrete motions and scores a path as the total number of motions that are invalid. The path is iteratively refined by replacing interior configurations with randomly perturbed configurations within a fixed distance λ; the start and goal states remain fixed. If the new path contains fewer colliding motions and, thus lower cost, it is kept and further refined in subsequent iterations.

To mitigate the path growing significantly longer than the original input path due to repeated perturbations, a shortcutting procedure (Geraerts and Overmars, 2007) is also applied to the perturbed path that connects non-consecutive configurations if this does not increase the total path cost. Finally, since the path cost is directly related to the number of discrete configurations, the path is interpolated before scoring to contain a minimum number of motions n.

Algorithm 2: GradientDescent

Input: λ: Maximum perturbation distance; n: Minimum number of path segments

$\pi^* \leftarrow \texttt{Interpolate}(\texttt{InitialPath}(), n)$
$c^* \leftarrow \texttt{PathCost}(\pi^*)$
while $c > 0$ **do**
  $\pi \leftarrow \texttt{Perturb}(\pi^*, \lambda)$
  $\pi \leftarrow \texttt{Interpolate}(\texttt{Shortcut}(\pi), n)$
  $c \leftarrow \texttt{PathCost}(\pi)$
  if $c < c^*$ **then**
    $\pi^* \leftarrow \pi$
    $c^* \leftarrow c$
return π^*

The gradient descent approach was evaluated on the steps in Figures 10 and 11 using a series of maximum perturbation distances λ, ranging from $\pi/24$ to $\pi/3$. During perturbation, each joint

Table 1: The median and interquartile range of a variety of metrics for each step when using gradient descent to find paths for R2. All values are taken over fifty independent runs. A timeout of 10 seconds is enforced in step 1, and 30 seconds for steps 2 and 3 in the 14-DoF experiments. Since the 21-DoF experiments induce a much larger search space, a timeout of 30 and 60 seconds is used for each step, respectively. **Declination:** maximum angular deviation of the torso from vertical along the solution path (degrees). **DT:** total Cartesian distance traveled through the workspace (cm) by a point mapped on the link noted. Time is measured in seconds. The best performances are marked in bold in each row.

		λ	$\pi/3$	$\pi/6$	$\pi/12$	$\pi/24$
14-DoFs	**Step 1**	**Time**	8 (8.3)	3 (5.2)	**1** (3.8)	2 (1.3)
		% Converged	34	**52**	24	16
		Declination	91 (37)	75 (60)	28 (24)	**28** (18)
		DT (head)	287 (340)	171 (210)	**124** (62)	129 (43)
		DT (waist)	156 (100)	105 (67)	**94** (63)	98 (48)
		DT (l-foot)	316 (480)	227 (290)	144 (190)	**116** (73)
	Step 2	**Time**	9 (8.0)	3 (4.1)	**2** (3.7)	6 (10)
		% Converged	**90**	78	46	24
		Declination	97 (37)	73 (28)	63 (40)	**46** (55)
		DT (head)	260 (220)	250 (200)	**236** (230)	241 (120)
		DT (waist)	168 (130)	**151** (100)	216 (130)	173 (100)
		DT (r-foot)	499 (200)	**433** (190)	497 (210)	509 (210)
	Step 3	**Time**	8 (10)	6 (5.4)	3 (3.4)	**3** (4.7)
		% Converged	**90**	78	30	22
		Declination	114 (32)	101 (32)	91 (22)	**59** (34)
		DT (head)	523 (120)	497 (230)	443 (240)	**328** (120)
		DT (waist)	227 (44)	227 (34)	222 (41)	**199** (32)
		DT (l-foot)	374 (210)	414 (230)	370 (120)	**362** (86)
21-DoFs	**Step 1**	**Time**	21 (7.1)	4 (5.8)	**3** (2.8)	4 (8.2)
		% Converged	28	**72**	58	30
		Declination	92 (45)	71 (29)	42 (22)	**30** (33)
		DT (head)	211 (170)	206 (110)	**183** (120)	195 (83)
		DT (waist)	142 (68)	128 (37)	124 (34)	**114** (38)
		DT (l-foot)	150 (140)	189 (110)	**168** (130)	178 (93)
		DT (r-palm)	215 (67)	**198** (67)	178 (110)	199 (98)
	Step 2	**Time**	25 (24)	9 (17)	10 (26)	**3** (19)
		% Converged	52	**66**	38	24
		Declination	103 (32)	79 (29)	75 (48)	**58** (16)
		DT (head)	317 (130)	326 (120)	291 (120)	**285** (69)
		DT (waist)	195 (69)	197 (56)	199 (43)	**189** (23)
		DT (r-foot)	450 (180)	427 (200)	**425** (260)	434 (120)
		DT (r-palm)	353 (94)	361 (140)	335 (190)	**308** (53)

can change position by no more than λ in a single step. The input path in each experiment is the straight-line configuration space path connecting the start to the goal, and each path is interpolated to a minimum of 20 states.

Summary statistics are presented in Table 1. These experiments show an intuitive trade-off between path length and time to the first solution, particularly as the dimensionality increases. When λ is large, it is possible to converge quickly to a collision free path, but this can significantly degrade the quality of the resulting path. The maximum declination angle of R2 uniformly decreases as λ shrinks, from more than 90 degrees with $\lambda = \pi/3$ to around 30 degrees in step 1 for $\lambda = \pi/24$ in both the 14 and 21-DoF joint sets, affirming the sensitivity of torso rotations to the motions of the legs. Analogously, the distance traveled by the head, waist, and free foot of R2 correlates with the magnitude of λ. The ability to converge to a solution quickly, however, inversely correlates with λ in the gradient descent method. In the most extreme case, the convergence rate drops from 90% to 22% on R2's 14-DoF step 3 as λ decreases from $\pi/3$ to $\pi/24$.

Another challenging phenomenon is controlling for the relative difficulty of a specific motion planning instance. Relatively short solution paths exist for step 1 in both the 14 and 21-DoF experiments, as well as the 21-Dof step 2 experiment. In these steps, the gradient descent approach has difficulty converging to a collision-free path for large values of λ because small motions are required. The time to converge is also substantially longer on these steps when $\lambda = \pi/3$, indicating that a single set of parameters is likely elusive in this domain.

6.1.2 Traditional sampling-based methods

The next set of experiments compares XXL with several existing sampling-based methods that return the first solution path found using a variety of informative heuristics. Specifically, XXL is compared to the canonical RRT and bi-directional RRT-CONNECT algorithms (LaValle and Kuffner, 2001) that utilize the notion of nearest neighbors in the configuration space, KPIECE and the bi-directional version BKPIECE (Şucan and Kavraki, 2012) representing projection-based algorithms the perform distance checks in a lower-dimensional Euclidean space, STRIDE (Gipson et al., 2013) which exhibits good scalability in planning instances for high-DoF manipulators by leveraging the internal structure of a nearest-neighbors data structure, and TSRRT (Shkolnik and Tedrake, 2009) which uses the workspace end-effector position of the unattached foot link to perform nearest-neighbor and distance queries. For the 14-DoF experiments, XXL utilizes a two-point PDG that projects the position of R2's waist and unattached foot. When the right arm is also included in the 21-DoF experiments, XXL utilizes a three-point PDG projecting the waist, unattached foot, and right hand. Workspace-guided sampling for XXL is implemented using a pseudoinverse-based iterative IK solver (Sciavicco and Siciliano, 1996). Solution paths from each method are smoothed in a post-processing step using a typical shortcutting procedure to shorten the path where possible (e.g., Geraerts and Overmars, 2007), provided that the maximum planning time has not yet elapsed.

Summary statistics for each of the steps for the planners listed above are presented in Table 2 for both the 14-DoF and 21-DoF experiments. The first two rows of each step show the median time and success rate of each of the methods over 50 independent trials. Virtually all methods are able to find a solution path consistently without exceeding the timeout. Naturally, the 21-DoF steps took longer time for most methods compared to the 14-DoF joint configurations. XXL is competitive in terms of solution times even with the additional overhead of the workspace-guided primitives.

It is worth noting that several of the canonical methods are able to return solution paths very quickly. RRT and RRT-CONNECT, for example, consistently find a solutions for each of the 14 and 21-DoF steps in around 1 second. The quality of these solutions, however, is often too poor to safely execute in practice even after applying standard shortcutting procedures (Geraerts and Overmars, 2007). Consider first the maximum declination angle, shown in the third row of each step in Table 2.

Table 2: The median and interquartile range of a variety of metrics for each step when planning for R2. All values are taken over fifty independent runs after applying standard path simplification procedures. A timeout of 10 seconds is enforced in step 1, and 30 seconds for steps 2 and 3 in the 14-DoF experiments. Since the 21-DoF experiments induce a much larger search space, a timeout of 30 and 60 seconds is used for each step, respectively. **Declination:** maximum angular deviation of the torso from vertical along the solution path (degrees). **DT:** total Cartesian distance traveled through the workspace (cm) by a point mapped on the link noted. Time is measured in seconds. The best performances are marked in bold in each row.

			BKPIECE	KPIECE	RRT	RRT-CNCT	STRIDE	TSRRT	XXL
14-DoFs	**Step 1**	**Time**	3 (3.0)	1 (1.0)	**0** (0.20)	1 (0.40)	1 (1.9)	2 (1.6)	1 (0.80)
		% Solved	94	100	100	100	100	100	100
		Declination	127 (16)	113 (59)	**4** (73)	108 (59)	109 (24)	29 (16)	16 (0.)
		DT (head)	505 (400)	428 (440)	**35** (330)	534 (390)	552 (270)	125 (74)	82 (8.0)
		DT (waist)	197 (130)	173 (120)	**34** (140)	211 (140)	215 (95)	80 (49)	74 (0.)
		DT (l-foot)	545 (440)	520 (360)	111 (350)	549 (350)	598 (230)	141 (150)	**65** (76)
	Step 2	**Time**	4 (1.8)	1 (1.3)	1 (0.50)	0 (0.50)	2 (2.4)	1 (0.80)	**0** (0.30)
		% Solved	100	100	100	100	100	100	100
		Declination	128 (20)	124 (24)	98 (49)	106 (34)	96 (37)	48 (14)	**20** (7.4)
		DT (head)	599 (410)	518 (310)	484 (400)	635 (290)	454 (310)	293 (90)	**179** (85)
		DT (waist)	279 (150)	253 (130)	245 (120)	287 (170)	260 (150)	182 (53)	**130** (58)
		DT (r-foot)	718 (370)	595 (410)	619 (300)	653 (320)	591 (210)	391 (150)	**278** (22)
	Step 3	**Time**	4 (2.7)	1 (2.5)	1 (0.60)	**0** (0.30)	3 (2.6)	1 (0.50)	8 (5.0)
		% Solved	100	100	100	100	100	100	100
		Declination	144 (16)	134 (26)	121 (49)	128 (48)	124 (29)	**50** (24)	74 (25)
		DT (head)	669 (290)	633 (350)	566 (260)	704 (260)	565 (190)	**307** (110)	452 (150)
		DT (waist)	311 (140)	297 (150)	272 (140)	301 (120)	269 (110)	**195** (47)	219 (68)
		DT (l-foot)	720 (290)	690 (270)	679 (320)	797 (510)	615 (380)	**389** (110)	417 (160)
21-DoFs	**Step 1**	**Time**	6 (4.2)	1 (0.90)	**1** (0.30)	1 (0.90)	2 (1.6)	2 (2.1)	3 (4.5)
		% Solved	100	100	100	100	100	100	100
		Declination	123 (21)	89 (40)	70 (39)	88 (47)	79 (34)	28 (15)	**19** (15)
		DT (head)	579 (460)	339 (220)	348 (130)	392 (380)	299 (230)	**147** (57)	161 (94)
		DT (waist)	229 (190)	167 (110)	173 (100)	197 (150)	136 (87)	**108** (32)	114 (53)
		DT (l-foot)	564 (560)	431 (290)	410 (280)	414 (420)	340 (270)	**200** (68)	237 (110)
		DT (r-palm)	550 (560)	387 (280)	357 (200)	449 (400)	340 (210)	**153** (59)	194 (110)
	Step 2	**Time**	6 (3.2)	4 (7.2)	1 (1.0)	**1** (1.4)	9 (8.9)	20 (16)	1 (3.0)
		% Solved	100	100	100	100	100	94	100
		Declination	127 (21)	125 (31)	99 (44)	116 (30)	111 (32)	**55** (22)	99 (0.)
		DT (head)	743 (430)	730 (310)	546 (220)	808 (300)	676 (450)	382 (250)	**324** (0.)
		DT (waist)	378 (240)	368 (140)	289 (97)	411 (150)	326 (180)	245 (97)	**206** (0.)
		DT (r-foot)	789 (390)	836 (320)	648 (340)	870 (480)	763 (460)	**401** (140)	492 (0.)
		DT (r-palm)	739 (520)	779 (230)	602 (270)	782 (370)	647 (410)	418 (180)	**292** (0.)

TSRRT and XXL, which utilize the task-space heuristic, are able to find paths that consistently reduce this measure compared to the other methods without explicitly optimizing this criteria; the workspace-guided sampling scheme that biases toward existing configurations (i.e., seed state) to discover new configurations. Random sampling can easily discover contorted configurations of R2, particularly when the hip-joint of the attached foot is moved.

The declination angle is just one measure of solution quality. The DT rows of each step in Table 2 measure the total distance traveled by a few points of interest on the robot on the final solution paths. A path with relatively low swept volume would induce small values for each of the points chosen. Highly undesirable motions can be identified by observing a large Cartesian distance for points on distal links of the robot (i.e., head, foot). In particular, a large head distance coupled with a high max declination angle is indicative of a somersault motion. Large distances for an unattached limb is indicative of a superfluous path. XXL is highly competitive in this measure across each of the five steps, consistently finding solutions with the shortest or near the shortest distance traveled in nearly all instances examined. TSRRT is similarly competitive in this measure.

The task-space projection used by both TSRRT and XXL becomes problematic for manipulators when the dimensionality grows, as illustrated in Figure 2. XXL combats this effect by disambiguating projections via the PDG, and this improvement is evidenced in 21-DoF step 2 which requires a complex coordination of the motion of three limbs within the ISS. Although TSRRT and XXL find paths with comparable quality, XXL consistently finds solutions up to 20x faster in this example and with substantially better quality compared to the other methods examined.

Finally, note from Table 2 that the variance of XXL's solution paths is near zero in certain cases. This is a property of the search when the start and goal states can be connected via a short path through the decomposition, sometimes within the same decomposition cell. The post-processing procedures employed are adept at removing artifacts from the random sampling process. This phenomenon is discussed further in Section 7.

6.1.3 Cost-aware sampling-based methods

In this section, XXL is compared with two popular, cost-aware sampling based algorithms when planning for R2 in the ISS using the same steps as in the previous experiments. Although XXL does not explicitly optimize for any particular solution criteria, the methods compared here are able to optimize for either the declination or the Cartesian distance metrics used in the previous section to discover more desirable solution paths. XXL is compared with TRRT (Jaillet et al., 2010; Devaurs et al., 2013), a method that probabilistically rejects sampled configurations that are considered high-cost. For a configuration with torso declination angle θ_d, the configuration cost used by TRRT is θ_d for $\theta \leq \pi/4$ and e^{θ_d} otherwise to encourage configurations with small declination angles. XXL is also compared with RRT* (Karaman and Frazzoli, 2011) which, unlike TRRT, optimizes over a whole-path metric. The objective for RRT* is to minimize the total Cartesian distance traveled through the workspace by all points in the PDG.

Summary statistics for each of the steps are presented in Table 3 for both the 14-DoF and 21-DoF R2 experiments. Although XXL is not strictly better than these methods when assessing the path quality, XXL is highly competitive over both quality metrics and is able to find solutions in an order of magnitude faster time in all five steps. RRT* always ran until the timeout was reached due to its asymptotic convergence properties. TRRT proved highly sensitive to parameter configurations that trade-off finding a solution quickly versus rejecting configurations that increase the declination angle. The Metropolis rejection criteria makes TRRT less likely to accept a configuration with a high declination angle as the search progresses, and such a configuration may be necessary to discover a solution path.

To a certain extent these experiments, along with those from Section 6.1.1, affirm the difficulty

Table 3: The median and interquartile range of a variety of metrics for each step when planning for R2 using XXL or optimizing planners. All values are taken over fifty independent runs after applying standard path simplification procedures. A timeout of 10 seconds is enforced in step 1, and 30 seconds for steps 2 and 3 in the 14-DoF experiments. Since the 21-DoF experiments induce a much larger search space, a timeout of 30 and 60 seconds is used for each step, respectively. **Declination:** maximum angular deviation of the torso from vertical (degrees). **DT:** total Cartesian distance traveled through the workspace (centimeters) by a point mapped on the link noted. Time is measured in seconds. The best performances are marked in bold in each row.

			RRT*	TRRT	XXL
14-DoFs	**Step 1**	**Time**	10 (0.)	6 (2.2)	**1** (0.80)
		% Solved	100	14	100
		Declination	**8** (5.7)	55 (44)	16 (0.)
		DT (head)	**56** (13)	330 (220)	82 (8.0)
		DT (waist)	**51** (7.1)	152 (98)	74 (0.)
		DT (l-foot)	65 (170)	332 (340)	**65** (76)
	Step 2	**Time**	30 (0.)	20 (9.0)	**0** (0.30)
		% Solved	100	36	100
		Declination	48 (33)	75 (38)	**20** (7.4)
		DT (head)	243 (90)	678 (240)	**179** (85)
		DT (waist)	138 (42)	302 (110)	**130** (58)
		DT (r-foot)	332 (73)	860 (280)	**278** (22)
	Step 3	**Time**	30 (0.10)	17 (6.3)	**8** (5.0)
		% Solved	100	98	100
		Declination	**70** (27)	92 (23)	74 (25)
		DT (head)	**361** (120)	791 (250)	452 (150)
		DT (waist)	**204** (24)	382 (130)	219 (68)
		DT (l-foot)	**405** (130)	955 (370)	417 (160)
21-DoFs	**Step 1**	**Time**	30 (0.10)	9 (9.0)	**3** (4.5)
		% Solved	100	100	100
		Declination	55 (39)	43 (17)	**19** (15)
		DT (head)	206 (120)	401 (210)	**161** (94)
		DT (waist)	**112** (35)	188 (68)	114 (53)
		DT (l-foot)	**220** (79)	513 (240)	237 (110)
		DT (r-palm)	233 (150)	438 (200)	**194** (110)
	Step 2	**Time**	60 (0.10)	46 (21)	**1** (3.0)
		% Solved	100	56	100
		Declination	**60** (28)	68 (25)	99 (0.)
		DT (head)	**317** (100)	728 (330)	324 (0.)
		DT (waist)	**194** (46)	414 (130)	206 (0.)
		DT (r-foot)	**413** (170)	893 (350)	492 (0.)
		DT (r-palm)	405 (140)	838 (310)	**292** (0.)

of defining a good optimization criteria for a complex and high-dimensional system like R2. In many instances, TRRT was either unable to find a solution within the timeout or could find a solution but was unable to consistently improve the declination angle. TRRT improves the declination angle in the solution path compared to the traditional sampling-based methods evaluated above, but XXL and in some cases RRT* finds solutions with an even smaller declination angle.

Finally, consider the Cartesian distances traveled by R2 in Table 3 from TRRT. These distances are 2-5x longer in these five steps compared not only to RRT*, but also many of the traditional sampling-based methods from Table 2. Minimizing declination angle alone can result in very long solution paths due to the difficulty of randomly sampling configurations with a small declination angle. Finding a solution path with zero torso declination is possible but requires solving an instance of motion planning on a *constraint manifold* (e.g., Berenson et al., 2011), a related but distinct planning problem. Note that minimizing Cartesian distance traveled correlates with a smaller torso declination angle in Table 3 but is not strictly the case. Consider 21-DoF Step 1. XXL finds solutions with similar Cartesian distance compared to RRT*, but the declination angle is almost 3x smaller with the XXL solution.

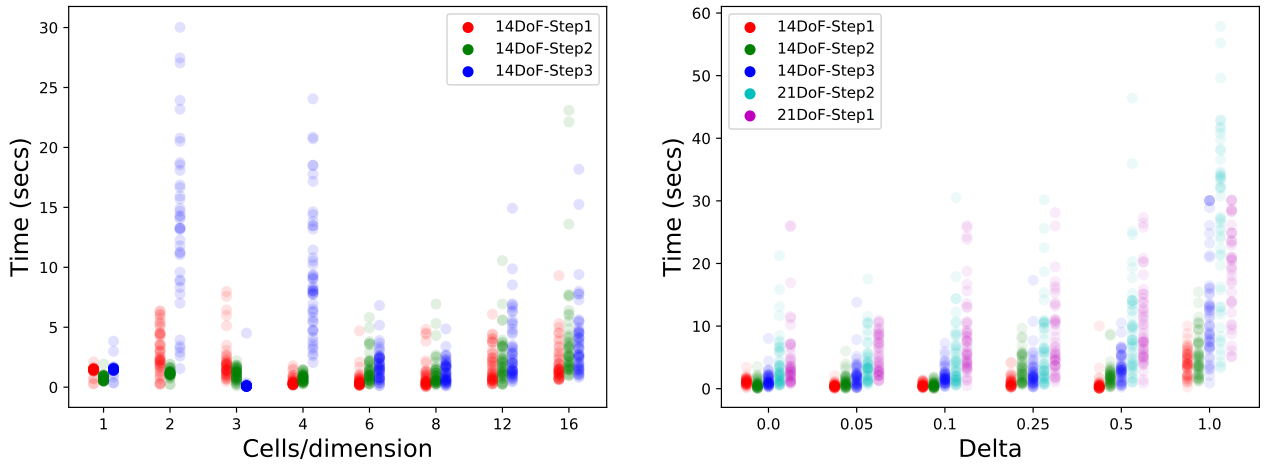
6.1.4 Sensitivity analysis

The sensitivity of XXL to its workspace decomposition is evaluated in this section. Recall that the workspace decomposition influences how XXL guides the search through the configuration space. Two of XXL's search parameters are examined when planning for R2 in the ISS: the resolution of the 3D workspace decomposition grid and the rate δ that XXL uses to compute the lead using a random walk.

In the first experiment, the resolution of the workspace decomposition in XXL is varied. Smaller grid cells allow for focused sampling and connection to occur between configurations that are likely to be close in the configuration space. This resolution comes at the expense of requiring XXL to connect configurations along each cell in the lead. The total number of grid cells in each dimension of the decomposition is varied from 1-16, translating to cell volumes that range from $\sim 4m^3$ with just one cell to $\sim 0.25m^3$ when there are 16 cells per dimension. For reference, XXL was evaluated using four cells per dimension in the prior experiments.

Figure 12a plots the distribution of the time to the first solution for XXL while varying the decomposition resolution in the three 14-DoF steps for R2 from Figure 10. Each configuration of problem and decomposition is simulated fifty times. Intuitively, the figure shows that a very coarse decomposition (2-3 cells per dimension) is generally deleterious to the search efficacy since the lead is not informative and sampling is not confined to small portions of the workspace. As the number of cells increases, a downward trend is observed in the distribution of times and their variances, reaching the floor around 4-6 cells per dimension depending on the experiment. As the resolution of the decomposition increase, the distribution of times begins to diffuse across all three steps. The one-cell decomposition is peculiar and we refer the reader to Section 7 for an extended discussion of this case. Note that 14-DoF step 3 proved difficult for the 2 and 4 cells/dimension configurations. This is attributable to the relatively small motion required, a singular joint configuration on R2's legs, and an inconvenient decomposition boundary between the start and goal configurations; the higher resolution decompositions are able to better navigate the joint singularity. The three-cell case is similar to the one-cell case in that there is no decomposition boundary that XXL must navigate across and it is possible to connect directly from start to goal within one cell given some lucky sampling.

In a second experiment, the workspace decomposition is fixed and the rate δ that XXL performs a random walk to compute the lead instead of a minimum cost path is varied between 0 and 1. When $\delta = 0$, XXL never uses the random walk and exclusively uses the learned costs to compute the lead. On the other hand XXL always uses the random walk and the workspace information accumulated during the search is never used when $\delta = 1$. For reference, XXL was evaluated using $\delta = 0.05$ in the prior experiments. There are six cells per dimension in these experiments and each configuration is



(a) XXL with variable resolution grid

(b) δ sensitivity in XXL

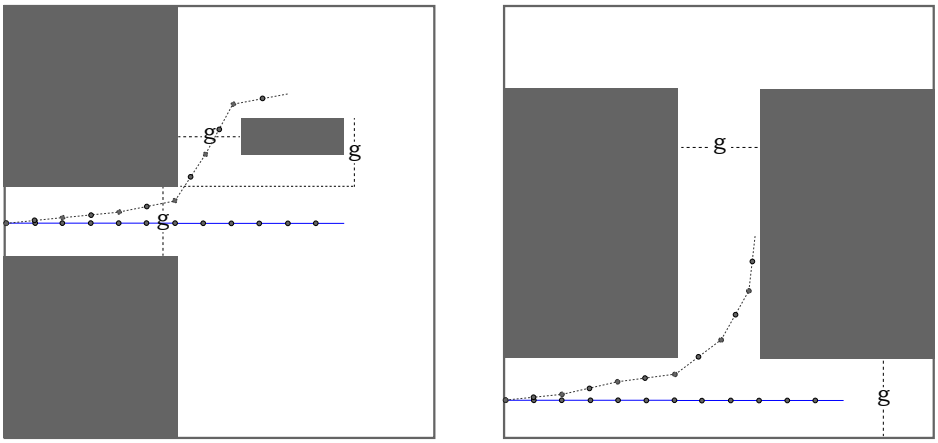
Figure 12: (left) The distribution of time to the first solution for XXL as the resolution of the workspace decomposition varies in the three 14-DoF steps for R2. (right) The distribution of time to the first solution for XXL with varying values of δ over the five R2 steps; at $\delta = 1$, XXL's lead computation is always a random walk. In each experiment, XXL is simulated fifty times.

simulated fifty times.

The distributions of time to the first solution over the five R2 steps as δ varies are plotted in Figure 12b. Naturally, these distributions show that large values of δ deteriorate XXL's ability to consistenly find a solution; the mean and variance across each of the five steps increases noticeably when δ is 0.25 or more. When no random walk is performed, XXL uses only information gained during the search to recompute the workspace lead and XXL does relatively well. Adding a small bit of randonmess to the lead computation ($\delta = 0.05$) generally improved XXL's ability to find a solution.

6.2 Planar kinematic chains

The generalizability and scalability of XXL is further assessed in a set of experiments that plan for a unit-length planar kinematic chain with a varying number of revolute joints. These problems represent particularly difficult instances of motion planning due to the high-dimensionality of the system being planned, the constrained nature of the underlying workspace, and that self-intersections of the chain are disallowed. In the first environment, the *constricted* world, the first half of the chain lies in a narrow gap between two obstacles and the end-effector must navigate a narrow passage, shown in Figure 13a. The second environment, named the *corridor* world, requires the entire chain to maneuver through a narrow corridor as shown in Figure 13b. The objective in both of these scenarios is for the end-effector to reach a predetermined position with any orientation. To prevent each scenario from becoming progressively easier as the number of joints in the unit-length chain grows (the link lengths shrink), obstacle sizes in these environments are a function of the number of joints. Figure 13 shows how obstacles in both scenarios change as the number of joints in the fixed-length chain grows. Computation time in the *constricted* scenario is limited to 180 seconds, and a timeout of 120 seconds is enforced for the *corridor* environment. For a planar kinematic chain with N joints, XXL utilizes a two-point PDG that projects the position of the midpoint and end-effector of the chain into a 2D grid decomposition of the workspace with $\frac{N}{3}$ cells in the *corridor* world and $\frac{N}{2}$ cells in the *constricted* world along each workspace dimension. Workspace-guided sampling and inverse kinematics for XXL and TSRRT is implemented using the FABRIK method (Aristidou and Lasenby, 2011).



(a) *Constricted* environment. The gap, g, is $\frac{2log(N)}{N}$, where N is the number of joints in the chain.

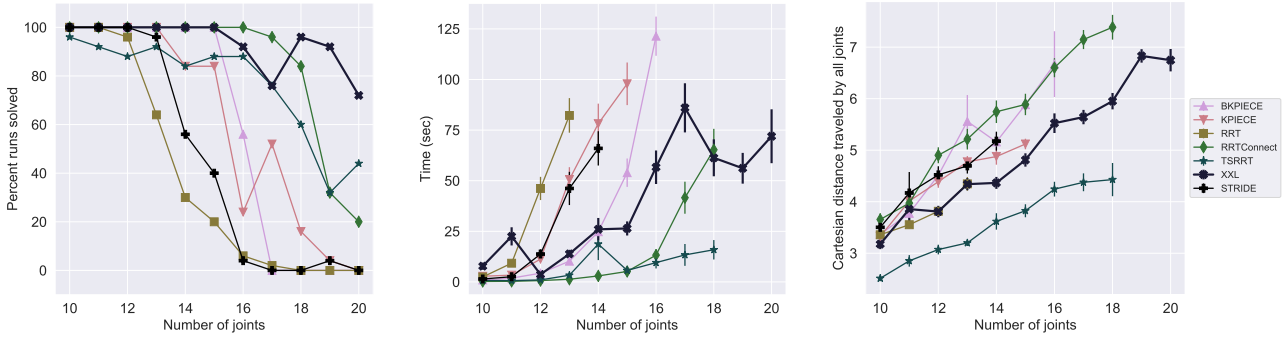
(b) *Corridor* environment. The gap, g, is a constant multiple of $\frac{logN}{N}$, where N is the number of joints in the chain.

Figure 13: The planning environments evaluated for a planar kinematic chain. The initial configuration of the chain is horizontal, and an example goal configuration is shown with the dashed chain. The objective is for the end-effector to reach a particular position (orientation is ignored).

Figure 14 shows the percentage of runs where a solution was found, the mean time to a solution, and the mean Cartesian distance traveled by all joints in the solution path for the *constricted* environment. XXL is the only method able to consistently find solutions as the problem scales to 19 and 20 joints. Although XXL is not the fastest method in the lower-dimensional instances, the computation times are competitive with several of the other planners. The final plot in Figure 14, the total Cartesian distance traveled by the chain, shows that XXL returns competitively short paths relative to most other methods evaluated as the number of joints increases. TSRRT was originally developed and evaluated in this planning domain, thus its relatively good performance is not unexpected. The single end-effector projection, however, does not scale well with the dimensionality of the chain and TSRRT has difficulties in finding solutions in the time allowed when the chain has 19 or more joints.

It is non-trivial to construct instances of planar manipulator planning where the problem is strictly more difficult as the number of joints are increased on a fixed length chain; different homotopy classes become *unlocked* depending on the size and location of the obstacles, as well as the link lengths in the chain. In this instance, the problem becomes noticeably more difficult at 16-17 joints and the difficulty is exaggerated in XXL by an unfortunate workspace decomposition containing cells that bisect the narrow workspace gap.

A similar set of results for the *corridor* environment is shown in Figure 15. None of the methods evaluated were able to consistently return solution paths in the time allowed after 27 joints with the exception of XXL. At the lower dimensions, XXL is highly competitive in terms of solution time as well as the Cartesian distance traveled, outperforming all but TSRRT in this measure after 20 joints. Similar to the prior experiment, the good performance of TSRRT is not unexpected here, although TSRRT did occasionally struggle in this instance even at the lowest dimensionalities; the end-effector projection is less informative when part of the chain is confined the narrow corridor. The ability to find a solution was problematic for TSRRT at the highest dimensions, however, demonstrating the additional gains achieved by XXL when employing multiple workspace projections.

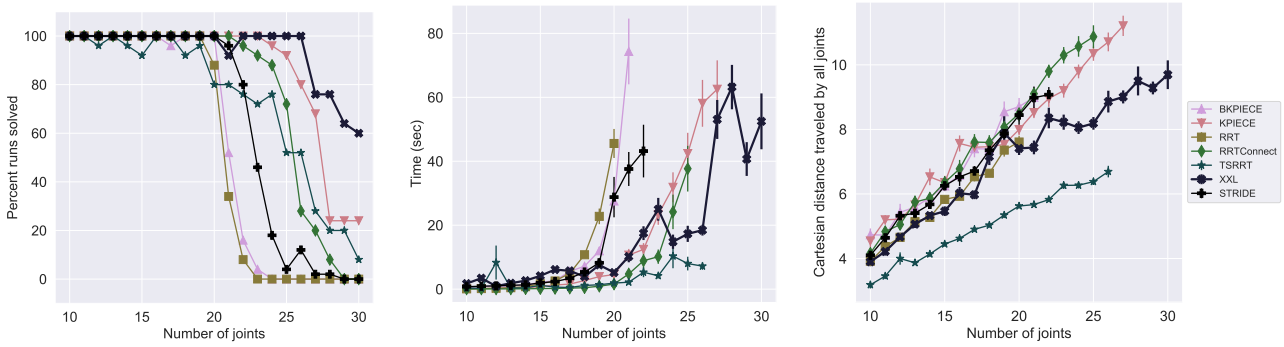


(a) Percent of runs where a solution is found

(b) Mean time to find a solution (secs)

(c) Cartesian distance traveled by all joints

Figure 14: Planner performance in the *constricted* scenario as the number of DoFs increases. Planning time is limited to 180 seconds, and all values are taken over 25 independent runs. Values for time and distance are omitted when the planner finds a solution in less than 50% of all runs. Error bars indicate the standard error of the mean. Cartesian distance is measured after applying path-simplification techniques.



(a) Solve percentage

(b) Mean solve time (secs)

(c) Cartesian distance traveled

Figure 15: Planner performance in the *corridor* scenario as the number of DoFs increases. Computation time is limited to 120 seconds, and all values are taken over 25 independent runs. Values for time and distance are omitted when the planner finds a solution in less than 50% of all runs. Error bars indicate the standard error of the mean. Cartesian distance is measured after applying path-simplification techniques.

7 Discussion: Guiding Heuristics in High-DoF Sampling-based Planning

The sampling-based motion planning literature is populated with many different heuristics used to guide the search through a potentially high-dimensional configuration space. Very little attention, however, has been given to the practical efficacy of these heuristics as the dimensionality of the underlying search space grows. This section presents an extended discussion on two well-known heuristics, the nearest neighbor of a configuration and the density of the search near a configuration, used to guide a sampling-based planner in discovering a solution path. Examples are given that show for particular instances of high-DoF manipulator planning in both 2D and 3D workspaces, increasing randomness in the search can perform just as well as algorithms that apply more sophisticated guiding heuristics according to the same path-quality measures used to evaluate XXL in the previous section. In particular, it is shown that a nearly random tree search can perform virtually identically to the well-known RRT algorithm and that removing the workspace decomposition from XXL yields a (nearly) random graph that can perform well in certain instances of planning for R2 and planar kinematic chains. The benefits of randomizing the neighbor selection have been demonstrated previously for the PRM (McMahon et al., 2012), showing that connecting to a random subset of the k-nearest neighbors can result in better connected roadmaps in a variety of planning queries with rigid and articulated systems.

7.1 Nearest-neighbor heuristic

The RRT algorithm pioneered the nearest-neighbor heuristic for sampling-based motion planning. In this seminal work, it is shown that selecting the nearest neighbor to a randomly sampled configuration for expansion naturally pushes the search into unexplored regions of the configuration space, the so-called *Voronoi bias* (LaValle and Kuffner, 2001). This work also anecdotally illustrates that for a simple 2D point system, a naïve tree search where a configuration is instead randomly chosen for expansion yields poor exploration the robot's configuration space.

Limiting the maximum length of an edge generalizes the notion of simulating the system under a particular control input for a discrete time interval during the search (LaValle and Kuffner, 2001). Note that this parameter does not appear explicitly in LaValle and Kuffner (2001) and must be inferred from the supporting text. Capping the maximum length of any one motion during the search has practical implications for many other sampling-based algorithms as well since the likelihood of the motion between two configurations being valid inversely correlates with the distance between them (Kavraki et al., 1996). Moreover, the time required to validate a motion is proportional to the length of the motion when using traditional local connector schemes. Although a discussion of maximum length does not appear explicitly in many works describing motion planning algorithms, this property is ubiquitous among single-query planners implemented in the Open Motion Planning Library (Şucan et al., 2012).

Given the well-known difficulties of defining a useful metric (Weber et al., 1998; Aggarwal et al., 2001; Cheng and LaValle, 2001) and fast nearest-neighbor searching (Zezula et al., 2006) in high-dimensional spaces, the *range-limited random tree* (RLRT) is proposed to evaluate the nearest-neighbor heuristic in the manipulator planning domain. Pseudocode for RLRT is given in Algorithm 3. The only difference between RLRT and a Canonical RRT is line 4: the configuration selected for expansion in the RLRT is chosen uniformly at random. A bidirectional version of RLRT can be formed by alternating the construction of two trees, rooted at the start and the goal. Joining the two trees is performed by attempting connections from the newly added configuration in one tree to $O(log\ N)$ randomly selected configurations in the other tree. Note that RLRT and BIRLRT still utilize a distance metric to limit the length of any given edge, but these algorithms significantly reduce their reliance on the metric since

the nearest configuration in the tree to one that is randomly sampled does not have to be evaluated during every iteration.

Algorithm 3: RLRT: Range-Limited Random Tree

```
(V, E) ← ({q_s}, ∅)
while KeepPlanning() do
    q_rand ← SampleValid()
    q_exp ← Random(V) // The lone difference
    d ← Distance(q_exp, q_rand)
    if d > ρ then
        q_rand ← Interpolate(q_exp, q_rand, ρ/d)
    if MotionValid(q_exp, q_rand) then
        V ← V ∪ {q_rand}
        E ← E ∪ {(q_exp, q_rand)}
return (V, E)
```

7.2 Density heuristic

Another popular heuristic to select a configuration for expansion is built upon the theory of planning in *expansive spaces* (Hsu et al., 1999). The key idea is to bias the search into unexplored areas of the space by selecting an existing configuration for expansion with probability inversely proportional to the *density* of existing nodes in the local neighborhood. As noted earlier in the related work, computing density in the configuration space suffers from many of the same pitfalls discussed earlier when utilizing nearest neighbors during the search. Instead, algorithms such as SBL (Sánchez and Latombe, 2001), KPIECE (Şucan and Kavraki, 2012), TSRRT (Shkolnik and Tedrake, 2009) and XXL combat the issue of metric generation by estimating the density near a particular configuration by projecting configurations into a low dimensional Euclidean space where distances are well defined; density is estimated by counting the number of configurations within a fixed radius or those that project into the same Euclidean grid cell.

Projecting configurations into a grid with one cell effectively negates the density heuristic since all configurations trivially fall into the same cell. A version of XXL where there is just one cell in the workspace decomposition, referred to as XXL1, is used to evaluate the efficacy of the workspace projection in guiding the search. In XXL1, the workspace lead is no longer informative since there is only one region in the decomposition. Moreover, the workspace-guided sampling procedure effectively becomes random since there are no specific workspace regions to restrict points on the robot to. As a result, XXL1 is similar to a PRM but with two key distinctions: any pair of nodes may be connected so long as a valid path exists between the nodes in the configuration space, and sampling is, generally speaking, no longer uniform over the configuration space due to the use of the workspace-guided sampler. PRM was also considered as another point for comparison in this set of experiments but is omitted due to low success rates in both the R2 and planar manipulator domains, a result of the limited computation time and expense of validating a dense set of motions.

7.3 Observations

In several instances of planning for high-DoF manipulators, it has been observed that the RLRT is statistically identical to the canonical RRT in the computation time and path-quality metrics. A similar observation also carries over to the bidirectional BIRLRT algorithm relative to RRT-CONNECT, and XXL1 relative to XXL.

Examples of these observations for the case of 14-DoF and 21-DoF planning for R2 are presented in Table 4. The table shows that in the second step of the 14-DoF example from the previous section that RLRT and RRT are indistinguishable in virtually all metrics evaluated. When inspecting the 21-DoF instance there is a bit more separation between RLRT and RRT, but the path quality measures are still statistically similar. A similar observation holds when comparing BIRLRT and RRT-CONNECT. Even though the path quality measures are statistically similar, paths from the canonical RRT-CONNECT algorithm are effectively the same as those deriving from a nearly random tree. XXL clearly outperforms XXL1 in several quality measures in the 14-DoF case, but this is not true in the 21-DoF instance. XXL1 shows much more variability in the solution paths, but tends to find paths with significantly less declination while staying competitive with XXL in the Cartesian distance measures.

These comparisons become more striking in the planar manipulator domain. Figure 16 shows the solve rate, mean time to solution, and total Cartesian distance traveled by each joint in the planar chain for the corridor environment (Figure 13b). From the plots, XXL1 and XXL perform comparably in terms of solve rate and time to solution until 28 joints where XXL1 sharply diverges and does not scale as well. XXL shows a slight edge over XXL1 in the Cartesian distance path measure. One surprising observation is that RLRT scales better than RRT in this instance. RRT shows a sharp drop in the solve rate after 20 joints, whereas a similar decline is not observed in RLRT until around 25 joints. BIRLRT performs comparably to RRT-CONNECT up to 22 joints in all three plots.

It must be stated that the observations described above are not necessarily indicative of any fundamental trends. The results presented in this section are not meant to be a comprehensive evaluation of the guiding heuristics employed by sampling-based planners. RLRT, BIRLRT, and XXL1 do not perform strictly better or worse than their counterparts that make heavier use of the nearest neighbor or density heuristics in the high-DoF manipulator planning scenarios attempted. In the 2D *constricted* environment, RRT and BIRLRT perform similarly to their more random counterparts according to Figure 17, although the fact that these methods are not better than random indicates that the efficacy of the nearest-neighbors heuristic for high-DoF problems is suspect.

Also from Figure 17, XXL1 scales noticeably better than XXL in the *constricted* world, finding solutions in consistently shorter times beginning at 12 joints. XXL1 is able to exploit the property in this problem that joints in the latter half of the problem are relatively unconstrained and the PDG decomposition allows for focused sampling of the joints in this part of the chain. XXL, on the other hand, must generate samples that project into and connect specific workspace cells and this task becomes challenging as the number of joints grows.

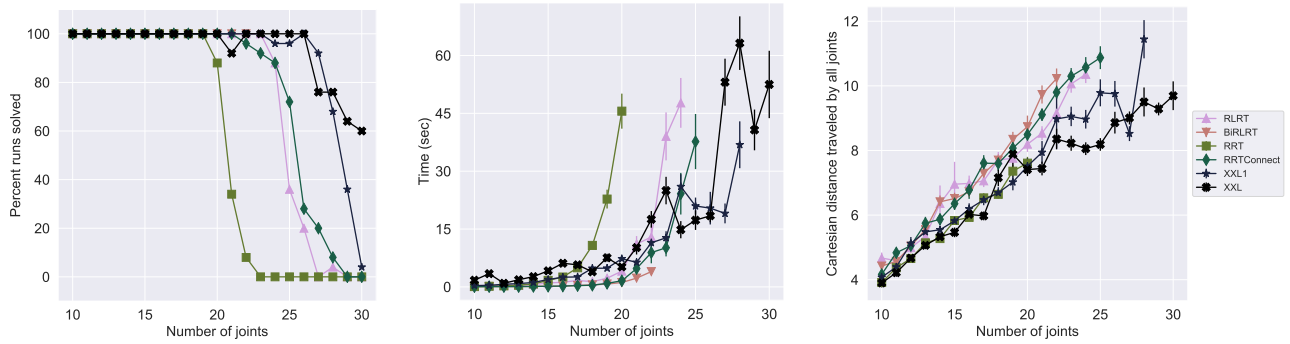
This discussion serves as a cautionary tale when using guiding heuristics in sampling-based motion planners as the dimensionality grows. One possibility for this effect is that the distance and density measures degrade in their ability to distinguish between configurations that may appear disparate in the workspace but are actually close together in the configuration space. Removing the bias from the search heuristics allows the search to more easily explore these motions. XXL attempts connections between configurations that project into the same decomposition cell, thus having large or even just one cell allows for this kind of exploration that may not have otherwise been considered. In short, if the guiding heuristic is truly reflective of the underlying problem, the search algorithm should be able to exploit this property. As the dimensionality of these problems increase, the same distance and density measures becomes less powerful in guiding the search in some cases.

8 Conclusion

This work presents a new probabilistically complete sampling-based algorithm, XXL, specially designed to plan the motions of high-DoF manipulator platforms to bring these systems to a pre-grasp pose. By utilizing the PDG to guide the motion of points of interest on the robot in conjunction with a novel

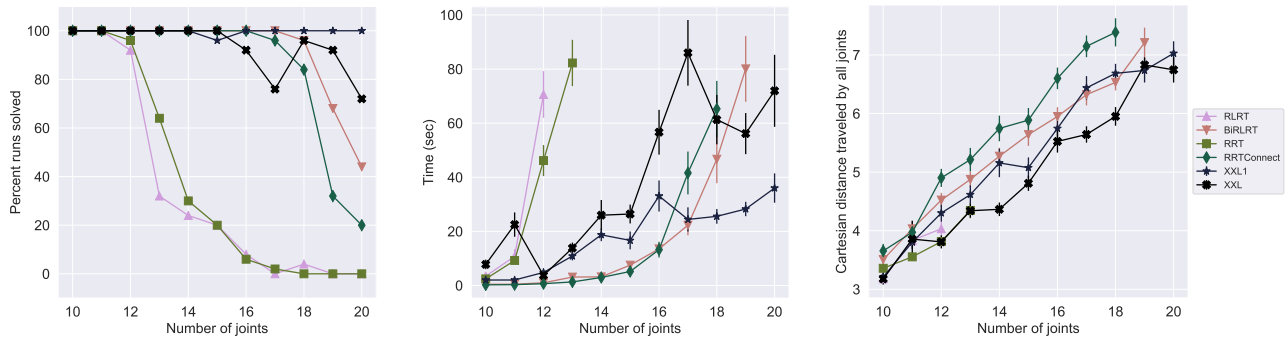
Table 4: The median and interquartile range of a variety of metrics for the second step of R2 inside the Destiny module, shown in Figures 10 and 11, respectively. One step plans for just the 14-DoF legs and the other plans for the 21-DoF legs and right arm. All values are taken over fifty independent runs after applying standard path simplification procedures. A 30 second timeout is enforced for the 14-DoF case, and 60 seconds is allotted for the 21-DoF step. **Declination:** maximum angular deviation of the torso from vertical (degrees). **DT:** total Cartesian distance traveled through the workspace (centimeters) by a point mapped on the link noted. Time is measured in seconds. The best performances for each pair of planning algorithms are marked in bold in each row.

		RLRT	RRT	BIRLRT	RRT-CONNECT	XXL1	XXL
14-DoF Step 2	**Time**	1 (2.1)	**1** (0.50)	**0** (0.30)	0 (0.50)	1 (0.10)	**0** (0.30)
	% Solved	100	100	100	100	100	100
	Declination	99 (36)	**98** (49)	110 (39)	**106** (34)	73 (42)	**20** (7.4)
	DT (head)	499 (320)	**484** (400)	**614** (220)	635 (290)	280 (86)	**179** (85)
	DT (waist)	**237** (120)	245 (120)	**279** (150)	287 (170)	149 (58)	**130** (58)
	DT (r-foot)	664 (380)	**619** (300)	706 (270)	**653** (320)	565 (290)	**278** (22)
21-DoF Step 2	**Time**	4 (5.3)	**1** (1.0)	**1** (1.1)	1 (1.4)	32 (14)	**1** (3.0)
	% Solved	100	100	100	100	14	100
	Declination	107 (28)	**99** (44)	**105** (27)	116 (30)	**25** (11)	99 (0.)
	DT (head)	685 (280)	**546** (220)	**703** (360)	808 (300)	331 (89)	**324** (0.)
	DT (waist)	337 (170)	**289** (97)	**382** (160)	411 (150)	224 (66)	**206** (0.)
	DT (r-foot)	728 (310)	**648** (340)	**773** (450)	870 (480)	530 (250)	**492** (0.)
	DT (r-palm)	675 (320)	602 (270)	696 (410)	782 (370)	394 (140)	**292** (0.)



(a) Solve percentage (b) Mean solve time (secs) (c) Cartesian distance traveled

Figure 16: Planner performance in the *corridor* scenario as the number of DoFs increases. Computation time is limited to 120 seconds, and all values are taken over 25 independent runs. Error bars indicate the standard error of the mean. Values for time and distance are omitted when the planner finds a solution in less than 50% of all runs. Cartesian distance is measured after applying path-simplification techniques.



(a) Solve percentage (b) Mean solve time (secs) (c) Cartesian distance traveled

Figure 17: Planner performance in the *constricted* scenario as the number of DoFs increases. Computation time is limited to 240 seconds, and all values are taken over 25 independent runs. Error bars indicate the standard error of the mean. Values for time and distance are omitted when the planner finds a solution in less than 50% of all runs. Cartesian distance is measured after applying path-simplification techniques.

workspace-guided sampling and connection strategy, XXL scales to realistic manipulator platforms with dozens of joints and obtains high-quality solutions relative to existing work in sampling-based motion planning. Simulated planning scenarios demonstrate that XXL exhibits competitive solution times relative to many other sampling-based algorithms in both the R2 and planar kinematic chain domains. Furthermore, XXL consistently returns solution paths of comparable or even superior quality when compared to cost-aware and asymptotically optimal methods without explicitly optimizing over any such criteria in substantially less computation time.

An extended discussion of the guiding heuristics used not only in XXL but also in many other sampling-based planning algorithms shows that care must be taken particularly in high-dimensional planning problems to ensure that the heuristic employed to help solve a particular problem actually aids in finding a solution. When the density heuristic used by XXL to guide the search is not informative, XXL still remains competitive in difficult instances of high-DoF manipulator planning.

It is important to note that XXL is not strictly better than any of the methods compared against. There exist planning queries in the humanoid and articulated robot space where a high-quality solution path is easy to find using a number of existing approaches. In particular are those where a relatively short, straight path in the configuration space is a valid solution. Generally speaking, XXL must perform more work to discover such solutions relative to many other sampling-based methods due to the overhead of the projection scheme and workspace decomposition, as well as the requirement that the path must explicitly pass through a contiguous set of workspace regions for at least one point in the PDG. XXL is a good candidate for instances of high-dimensional and whole-body manipulator planning, particularly for tree-like systems where subsets of the DoFs may not be especially important in achieving a good quality solution. This is evident in the 21-DoF R2 configurations where joints in R2's arm do not need to move while a leg is transitioning between a handrail. In the second, more complex step in these experiments, XXL found solutions where the hand moved between 1-5m less through the workspace to arrive at the goal configuration compared to every other method evaluated.

There are a number of potential future directions for this work. The connection strategy used by XXL and many other sampling-based methods can be adapted using techniques from previous works to derive an *anytime* algorithm that refines an initial solution path while computation time allows, often with provable guarantees of *almost sure* convergence to an optimal solution path. One of the most straightforward extensions is the computation of paths that minimize an arbitrary cost function

by annotating the edges of the roadmap with a real-value indicating the cost of each motion. The connection strategies employed by PRM* (Karaman and Frazzoli, 2011) or Anytime-TRRT (Devaurs et al., 2014) can be directly applied to XXL to obtain a method that globally optimizes over the cost of the entire path or locally minimizes the cost of each edge in the roadmap, respectively.

XXL could also be incorporated into a planner *portfolio* used to solve a planning problem when the difficulty is unknown. Portfolio frameworks may also be used in an *anytime* fashion where solution paths from one or more different planners are combined (hybridized) to form a new and improved solution path (Luna et al., 2013). Empirical results show that this kind of framework can converge much faster to a high-quality solution path for high-dimensional systems when compared to asymptotically-optimal methods (Luo and Hauser, 2014). Injecting the randomness from RLRT and related methods that reduce dependence on guiding heuristics may also be interesting to evaluate in the context of a hybridizing framework for diversity of solution paths.

When the robot has a closed kinematic chain, either when the robot contains two links whose parent is the other or as a result of a task like dual-arm manipulation, the PDG is no longer a directed, acyclic graph (DAG) and a topological ordering of the PDG does not exist. It is possible to relax the DAG requirement of the PDG to allow circular dependencies by collapsing the points in the PDG involved in a circular dependency into one super-node. The closure constraints of the super-node can then be resolved by XXL during sampling using standard techniques (LaValle, 2006).

Incorporating task-level constraints into XXL is another possible extension of this work. Such constraints impose restrictions on the valid configurations robot to generate paths with particular characteristics such as ensuring contact with a surface (Hauser et al., 2008) or maintaining the orientation of an object that is grasped by the manipulator (Cohen et al., 2014). A related topic involves planning the motions of legged humanoids and related dynamically-stabilized platforms. Sampling of and connection between dynamically-stable configurations can be achieved through precomputation (Burget et al., 2013) and even at runtime using inverse kinematics and gradient descent (Berenson et al., 2011) or by orthogonally projecting configurations onto the lower-dimensional constraint manifold embedded in the configuration space (Jaillet and Porta, 2013; Hauser, 2014) in conjunction with XXL's workspace guidance strategies.

Funding

R. Luna is supported by a NASA Space Technology Research Fellowship. L.E. Kavraki and M. Moll are supported in part by NSF CCF-1514372, NSF CCF-1139011, NSF IIS-1317849, and NASA NNX15AI58G.

References

Achlioptas D (2001) Database-friendly random projections. In: *ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*. pp. 274–281.

Aggarwal CC, Hinneburg A and Keim DA (2001) On the surprising behavior of distance metrics in high dimensional space. In: *8th International Conference on Database Theory*, *Lecture Notes in Computer Science*, volume 1973. Springer Berlin Heidelberg, pp. 420–434.

Ailon N and Chazelle B (2009) The fast Johnson-Lindenstrauss transform and approximate neighbors. *SIAM Journal on Computing* 39: 302–322.

Andoni A, Indyk P, Nguyen HL and Razenshteyn I (2014) Beyond locality-sensitive hashing. In: *The Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*. pp. 1018–1028.

Aristidou A and Lasenby J (2011) FABRIK: A fast, iterative solver for the inverse kinematics problem. *Graphical Models* 73(5): 243–260.

Badger J, Hulse A, Taylor R, Curtis A, Gooding D and Thackston A (2013) Model-based robotic dynamic motion control for the robonaut 2 humanoid robot. In: *IEEE-RAS International Conference on Humanoid Robots*. pp. 62–67.

Berenson D, Srinivasa S and Kuffner J (2011) Task space regions: A framework for pose-constrained manipulation planning. *International Journal of Robotics Research* 30(12): 1435–1460.

Bertram D, Kuffner J, Dillmann R and Asfour T (2006) An integrated approach to inverse kinematics and path planning for redundant manipulators. In: *IEEE International Conference on Robotics and Automation*. pp. 1874–1879.

Brock O and Kavraki LE (2001) Decomposition-based motion planning: A framework for real-time motion planning in high-dimensional configuration spaces. In: *IEEE International Conference on Robotics and Automation*. pp. 1469–1475.

Burget F, Hornung A and Bennewitz M (2013) Whole-body motion planning for manipulation of articulated objects. In: *IEEE International Conference on Robotics and Automation*. pp. 1656–1662.

Cheng P and LaValle SM (2001) Reducing metric sensitivity in randomized trajectory design. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. pp. 43–48.

Choset H, Lynch KM, Hutchinson S, Kantor G, Burgard W, Kavraki LE and Thrun S (2005) *Principles of Robot Motion: Theory, Algorithms, and Implementations*. MIT Press.

Cohen B, Chitta S and Likhachev M (2014) Single- and dual-arm motion planning with heuristic search. *International Journal of Robotics Research* 33(2): 305–320.

Devaurs D, Siméon T and Cortés J (2013) Enhancing the transition-based RRT to deal with complex cost spaces. In: *IEEE International Conference on Robotics and Automation*. pp. 4120–4125.

Devaurs D, Siméon T and Cortés J (2014) Efficient sampling-based approaches to optimal path planning in complex cost spaces. In: *Algorithmic Foundation of Robotics XI*. pp. 143–159.

Diankov R, Ratliff N, Ferguson D, Srinivasa S and Kuffner J (2008) Bispace planning: Concurrent multi-space exploration. In: *Robotics: Science and Systems IV*.

Diftler M, Ahlstrom T, Ambrose R, Radford N, Joyce C, Pena NDL, Parsons A and Noblitt A (2012) Robonaut 2 - initial activities on-board the ISS. In: *IEEE Aerospace Conference*. pp. 1–12.

Diftler M, Mehling J, Abdallah M, Radford N, Bridgwater L, Sanders A, Askew R, Linn D, Yamokoski J, Permenter F, Hargrave B, Platt R, Savely R and Ambrose R (2011) Robonaut 2 - the first humanoid robot in space. In: *IEEE International Conference on Robotics and Automation*. pp. 2178–2183.

Dragan AD, Lee KC and Srinivasa SS (2013) Legibility and predictability of robot motion. In: *ACM/IEEE International Conference on Human-Robot Interaction*. pp. 301–308.

Foskey M, Garber M, Lin MC and Manocha D (2001) A voronoi-based hybrid motion planner. In: *IEEE International Conference on Intelligent Robots and Systems*. pp. 55–60.

Geraerts R and Overmars MH (2007) Creating high-quality paths for motion planning. *International Journal of Robotics Research* 26(8): 845–863.

Gipson B, Moll M and Kavraki LE (2013) Resolution independent density estimation for motion planning in high-dimensional spaces. In: *IEEE International Conference on Robotics and Automation*. pp. 2437–2443.

Hastings WK (1970) Monte carlo sampling methods using markov chains and their applications. *Biometrika* 57(1): 97–109.

Hauser K (2014) Fast interpolation and time-optimization with contact. *International Journal of Robotics Research* 33(9): 1231–1250.

Hauser K, Bretl T, Latombe JC, Harada K and Wilcox B (2008) Motion planning for legged robots on varied terrain. *International Journal of Robotics Research* 27(11-12): 1325–1349.

Holleman C and Kavraki LE (2000) A framework for using the workspace medial axis in PRM planners. In: *IEEE International Conference on Robotics and Automation*. pp. 1408–1413.

Hsu D, Latombe JC and Motwani R (1999) Path planning in expansive configuration spaces. *International Journal of Computational Geometry and Applications* 9(4–5): 495–512.

Jaillet L, Cortés J and Siméon T (2010) Sampilng-based path planning on configuration space costmaps. *IEEE Transactions on Robotics* 26(4): 635–646.

Jaillet L and Porta JM (2013) Path planning under kinematic constraints by rapidly exploring manifolds. *IEEE Transactions on Robotics* 29: 105–117.

Johnson WB and Lindenstrauss J (1984) Extensions of lipschitz mappings into a hilbert space. *Contemporary Mathematics* 26: 189–206.

Kalakrishnan M, Chitta S, Theodorou E, Pastor P and Schaal S (2011) STOMP: Stochastic trajectory optimization for motion planning. In: *IEEE International Conference on Robotics and Automation*. pp. 4569–4574.

Karaman S and Frazzoli E (2011) Sampling-based algorithms for optimal motion planning. *International Journal of Robotics Research* 30(7): 846–894.

Kavraki LE, Kolountzakis MN and Latombe JC (1998) Analysis of probabilistic roadmaps for path planning. *IEEE Transactions on Robotics and Automation* 14(1): 166–171.

Kavraki LE, Švestka P, Latombe JC and Overmars MH (1996) Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation* 12(4): 566–580.

Kobilarov M (2011) Cross-entropy randomized motion planning. In: *Robotics: Science and Systems VII*.

Kuffner Jr JJ, Kagami S, Nishiwaki K, Inaba M and Inoue H (2002) Dynamically-stable motion planning for humanoid robots. *Autonomous Robots* 12: 105–118.

Kurniawati H and Hsu D (2004) Workspace importance sampling for probabilistic roadmap planning. In: *IEEE International Conference on Intelligent Robots and Systems*.

Kurniawati H and Hsu D (2008) Workspace-based connectivity oracle: An adaptive sampling strategy for PRM planning. In: *Algorithmic Foundation of Robotics VII*, volume 47. pp. 35–41.

Ladd AM and Kavraki LE (2004) Measure theoretic analysis of probabilistic path planning. *IEEE Transactions on Robotics and Automation* 20(2): 229–242.

Ladd AM and Kavraki LE (2005) Motion planning in the presence of drift, underactuation and discrete system changes. In: *Robotics: Science and Systems*.

LaValle SM (2006) *Planning Algorithms*. Cambridge University Press.

LaValle SM and Kuffner JJ (2001) Randomized kinodynamic planning. *International Journal of Robotics Research* 20(5): 378–400.

Likhachev M and Ferguson D (2009) Planning long dynamically feasible maneuvers for autonomous vehicles. *International Journal of Robotics Research* 28(8): 933–945.

Luna R, Şucan I, Moll M and Kavraki LE (2013) Anytime solution optimization for sampling-based motion planning. In: *IEEE International Conference on Robotics and Automation*. pp. 5053–5059.

Luo J and Hauser K (2014) An empirical study of optimal motion planning. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. pp. 1761–1768.

McMahon T, Jacobs S, Boyd B, Tapia L and Amato NM (2012) Local randomization in neighbor selection improves PRM roadmap quality. In: *IEEE International Conference on Intelligent Robots and Systems*. pp. 4441–4448.

Metropolis N, Rosenbluth AW, Rosenbluth MN and Teller AH (1953) Equation of state calculations by fast computing machines. *Chemical Physics* 21(6): 1087–1092.

Murray RM, Li Z and Sastry SS (1994) *A Mathematical Introduction to Robotic Manipulation*. CRC Press.

Persson SM and Sharf I (2014) Sampling-based A* algorithm for robot path-planning. *International Journal of Robotics Research* 33(13): 1683–1708.

Plaku E (2015) Region-guided and sampling-based tree search for motion planning with dynamics. *IEEE Transactions on Robotics* 31(3): 723–735.

Plaku E, Kavraki LE and Vardi MY (2010) Motion planning with dynamics by a synergistic combination of layers of planning. *IEEE Transactions on Robotics* 26(3): 469–482.

Quigley M, Gerkey B, Conley K, Faust J, Foote T, Leibs J, Berger E, Wheeler R and Ng A (2009) ROS: an open-source Robot Operating System. In: *ICRA Workshop on Open Source Software*.

Quinlan S and Khatib O (1993) Elastic bands: Connecting path planning and control. In: *IEEE International Conference on Robotics and Automation*. pp. 802–807.

Rickert M, Sieverling A and Brock O (2014) Balancing exploration and exploitation in sampling-based motion planning. *IEEE Transactions on Robotics* 30(6): 1305–1317.

Rodriguez S, Thomas S, Pearce R and Amato NM (2006) RESAMPL: A region-sensitive adaptive motion planner. In: *Algorithmic Foundations of Robotics VII*. pp. 285–300).

Sánchez G and Latombe JC (2001) A single-query bi-directional probabilistic roadmap planner with lazy collision checking. In: *The Tenth International Symposium on Robotics Research*. pp. 403–417.

Schulman J, Duan Y, Ho J, Lee A, Awwal I, Bradlow H, Pan J, Patil S, Goldberg K and Abbeel P (2014) Motion planning with sequential convex optimization and convex collision checking. *International Journal of Robotics Research* 33(9): 1251–1270.

Sciavicco L and Siciliano B (1996) *Modeling and Control of Robot Manipulators*. McGraw-Hill.

Shkolnik A and Tedrake R (2009) Path planning in 1000+ dimensions using a task-space voronoi bias. In: *IEEE International Conference on Robotics and Automation*. pp. 2061–2067.

Sisbot EA, Marin LF and Alami R (2007) Spatial reasoning for human robot interaction. In: *IEEE International Conference on Intelligent Robots and Systems*. pp. 2281–2287.

Şucan IA and Chitta S (2012) MoveIt! `http://moveit.ros.org`.

Şucan IA and Kavraki LE (2009) On the performance of random linear projections for sampling-based motion planning. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. pp. 2434–2439.

Şucan IA and Kavraki LE (2012) A sampling-based tree planner for systems with complex dynamics. *IEEE Transactions on Robotics* 28(1): 116–131.

Şucan IA, Moll M and Kavraki LE (2012) The open motion planning library. *IEEE Robotics Automation Magazine* 19(4): 72–82. DOI:10.1109/MRA.2012.2205651.

Tang X, Thomas S, Coleman P and Amato NM (2010) Reachable distance space: Efficient sampling-based planning for spatially constrained systems. *International Journal of Robotics Research* 29(7): 916–934.

Toussaint M (2009) Robot trajectory optimization using apprroximate inference. In: *26th Annual International Conference on Machine Learning*. pp. 1049–1056.

van den Berg JP and Overmars MH (2005) Using workspace information as a guide to non-uniform sampling in probabilistic roadmap planners. *International Journal of Robotics Research* 24(12): 1055—-1071.

Wang LCT and Chen CC (1991) A combined optimization method for solving the inverse kinematics problem of mechanical manipulators. *IEEE Transactions on Robotics and Automation* 7(4): 489–499.

Weber R, Schek HJ and Blott S (1998) A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. In: *24th International Conference on Very Large Data Bases*. pp. 194–205.

Weghe MV, Ferguson D and Srinivasa SS (2007) Randomized path planning for redundant manipulators without inverse kinematics. In: *IEEE-RAS International Conference on Humanoid Robots*. pp. 477–482.

Wilmarth SA, Amato NM and Stiller PF (1999) MAPRM: A probabilistic roadmap planner with sampling on the medial axis of the free space. In: *IEEE International Conference on Robotics and Automation*. pp. 1024–1031.

Yang Y and Brock O (2005) Efficient motion planning based on disassembly. In: *Robotics: Science and Systems*.

Yeh HYC, Denny J, Lindsey A, Thomas SL and Amato NM (2014) UMAPRM: Uniformly sampling the medial axis. In: *IEEE International Conference on Robotics and Automation*. pp. 90–97.

Yoshida E, Poirier M, Laumond JP, Kanoun O, Lamiraux F, Alami R and Yokoi K (2010) Pivoting based manipulation by a humanoid robot. *Autonomous Robots* 28: 77–88.

Zezula P, Amato G, Dohnal V and Batko M (2006) *Similarity Search: The Metric Space Approach*. Springer.

Zucker M, Ratliff N, Dragan AD, Pivtoraiko M, Klingensmith M, Dellin CM, Bagnell AJ and Srinivasa SS (2013) CHOMP: Covariant Hamiltonian optimization for motion planning. *International Journal of Robotics Research* 32(9–10): 1164–1193.