Article

The International Journal of Robotics Research
2018, Vol. 37(7) 779–817

DOI: 10.1177/0278364918779555
journals.sagepub.com/home/ijr

Sampling-based motion planning with reachable volumes for high-degree-of-freedom manipulators

Troy McMahon, Shawna Thomas and Nancy M Amato

Abstract

Motion planning for constrained systems is a version of the motion planning problem in which the motion of a robot is limited by constraints. For example, one can require that a humanoid robot such as a PR2 remain upright by constraining its torso to be above its base or require that an object such as a bucket of water remain upright by constraining the vertices of the object to be parallel to the robot's base. Grasping can be modeled by requiring that the end effectors of the robot be located at specified handle positions. Constraints might require that the robot remain in contact with a surface, or that certain joints of the robot remain in contact with each other (e.g., closed chains). Such problems are particularly difficult because the constraints form a manifold in C-space, and planning must be restricted to this manifold. High-degree-of-freedom motion planning and motion planning for constrained systems has applications in parallel robotics, grasping and manipulation, computational biology and molecular simulations, and animation. We introduce a new concept, reachable volumes, that are a geometric representation of the regions the joints and end effectors of a robot can reach, and use it to define a new planning space called RV-space where all points automatically satisfy a problem's constraints. Visualizations of reachable volumes can enable operators to see the regions of workspace that different parts of the robot can reach. Samples and paths generated in RV-space naturally conform to constraints, making planning for constrained systems no more difficult than planning for unconstrained systems. Consequently, constrained motion planning problems that were previously difficult or unsolvable become manageable and in many cases trivial. We introduce tools and techniques to extend the state-of-the-art sampling-based motion planning algorithms to RV-space. We define a reachable volumes sampler, a reachable volumes local planner, and a reachable volumes distance metric. We showcase the effectiveness of RV-space by applying these tools to motion planning problems for robots with constraints on the end effectors and/or internal joints of the robot. We show that RV-based planners are more efficient than existing methods, particularly for higher-dimensional problems, solving problems with 1000 or more degrees of freedom for multi-loop and tree-like linkages.

Keywords

motion planning, manipulators, constrained systems

1. Introduction

Constrained motion planning places constraints on the motion of an object (robot) and has applications in parallel robotics (Merlet, 2002), grasping and manipulation (Oriolo and Mongillo, 2005), computational biology and molecular simulations (Cortés and Siméon, 2005), and animation (Kallmann et al., 2003). Constraints can be used to model a wide variety of constrained systems. For example, one can require that a humanoid robot such as a PR2 (see http://www.willowgarage.com) remain upright by constraining its torso to be above its base or require that an object such as a bucket of water remain upright by constraining the vertices of the object to be parallel to the robot's base. Grasping can be modeled by requiring that the end effectors of the robot be located at specified handle positions. Constraints might also require that the robot remain in contact with a surface, or that certain joints of the robot remain in contact with each other (e.g., closed chains). Such constraints could be used in industrial automation to constrain a tool mounted on a robot to a surface or a seam (for example, we could constrain a welder mounted on a robot to a seam that needs to be welded). They could also be used to simulate contacts or binding in protein folding simulations.

Parasol Lab, Department of Computer Science and Engineering, Texas A&M University, College Station, TX, USA

Corresponding author:
Troy McMahon, Parasol Lab, Department of Computer Science and Engineering, Texas A&M University, 3112 TAMU, College Station, TX 77843-3112, USA.
Email: troymcmahon1@gmail.com

Sampling-based motion planning methods such as the graph-based probabilistic roadmap method (PRM) (Kavraki et al., 1996) and the tree-based rapidly exploring random tree (RRT) (LaValle and Kuffner, 1999) are state-of-the-art solutions to traditional motion planning problems. Unfortunately, these methods are poorly suited for many constrained problems where the constraints define a manifold in C-space and planning must be restricted to this manifold (LaValle et al., 1999). Previous methods have developed specialized samplers that generate samples satisfying constraints (Han and Amato, 2001; Han and Rudolph, 2007b; Tang et al., 2010) that can be used in combination with existing PRM-based methods to solve problems with constraints. However, these methods are either unable to handle high-degree-of-freedom (high-dof) systems or unsuited for systems with spherical or prismatic joints or systems that combine different types of joints.

We propose a new concept, *reachable volumes*, that are a geometric representation of the regions the joints and end effectors of a robot can reach, and use it to define a new planning space called RV-space where all points in RV-space automatically satisfy a problem's constraints. Samples and paths generated in RV-space naturally conform to constraints, making planning for constrained systems no more difficult than planning for unconstrained systems. Consequently, constrained motion planning problems that were previously difficult or unsolvable become manageable and in some cases trivial.

We define the reachable volume of a joint/end effector to be the volume of RV-space it can reach while satisfying a problem's constraints. The reachable volumes of a robot then are simply the collection of each joint's individual reachable volume. Reachable volumes generalize the concept of reachable distances (Tang et al., 2010) so that they can be applied to linkages, closed chains, and tree-like robots with prismatic and spherical as well as planar joints. Visualizations of reachable volumes have application in robot design where they allow designers to determine whether a robot can reach the areas it needs to perform the required tasks. They also have application in robot control where they allow an operator to determine what they can reach from a specified position, which can help them to decide where they should position the robot.

We introduce tools and techniques to extend the state-of-the-art sampling-based motion planning algorithms to RV-space. We propose a reachable volumes sampler, a reachable volumes local planner, and a reachable volumes distance metric. Reachable volumes sampling generates samples by iteratively sampling the joints of a robot in their individual reachable volume, resulting in samples that are guaranteed to satisfy a problem's constraints. Reachable volumes-based planners can solve problems with constraints applied to any combination of joints/end effectors, whereas most other methods (e.g. Han and Amato, 2001; Tang et al., 2010; Wang and Chen, 1991) assume a single constraint, usually on one of the end effectors. The reachable volumes local planner and distance metric can be used to generate constraint-satisfying local paths, even in problems such as closed chains where the constraints form a manifold. As part of the reachable volumes local planner, we present a novel method for stepping reachable volumes samples to generate samples that are close to the original while ensuring they satisfy the problem's constraints.

We show that the geometric complexity of a reachable volume is $O(1)$ in unconstrained problems as well as for many constrained problems. This allows us to generate samples in linear time with respect to the number of bodies in the robot, which is the best possible complexity for a sampler. In problems with more complex constraints, we present an $O(|L|^2\ |S|\ C(S))$ method for generating samples, where S is the set of constraints, $C(S)$ is the complexity of the constraints, and $|L|$ is the number of bodies in the robot. We also show that the reachable volumes of all of the joints/end effectors in a robot can be computed in $O(|J|\text{diameter}(R))$ time, where $|J|$ is the number of joints in the robot and diameter(R) is the robot's diameter (i.e. the longest distance between any two pairs of joints). This is superior to $O(|J|^2)$ time that would be required to compute these reachable volumes separately. Finally, we show that roadmaps generated using reachable volume sampling are probabilistically complete.

We present extensive experimental validation of sampling-based motion planning with reachable volumes. Our results show that reachable volumes sampling produces more valid samples than existing methods, that reachable volumes samples are easier to connect than other samples, and that reachable volumes sampling is more efficient at solving high-dimensional problems than existing methods. They also confirm that its running time is linear with respect to the number of bodies in the robot. We also show that the reachable volumes local planner can produce constraint-satisfying local paths with little overhead compared with the commonly used straight line local planner, which cannot find constraint-satisfying paths.

The main contributions of this work include the following.

- The *reachable volumes* concept, which denotes the volume of space that the joints and end effectors can reach while satisfying the problem constraints, and a new planning space called RV-space where all points automatically satisfy the constraints.
- Tools needed for sampling-based motion planning in RV-space including a reachable volumes sampler, a reachable volumes local planner, and a reachable volumes distance metric.
- Empirical evaluation of reachable volumes over a wide variety of systems including chains, closed chains, and tree-like robots with as many as 1,034 dofs.

The paper is organized in the following manner. In Section 2 we give an overview of related work and in Section 3 we define the set of problems that our work addresses. In

Section 4 we introduce the concept of *reachable volumes* and show how to compute them. In Section 5 we present primitive operations including a reachable volumes sampler, a reachable volumes local planner, and a reachable volumes distance metric. In Section 6 we present a reachable volumes planner that uses these operations. In Section 7 we evaluate sampling-based motion planning with reachable volumes.

This journal paper includes extended discussions of the material presented in McMahon et al. (2014b), McMahon et al. (2014a), and McMahon et al. (2015). It provides a more complete and mature handling of reachable volumes and includes additional motion planning primitives, a novel method for simultaneously computing the reachable volumes of all the joints and end effectors, a description of how to transform RV-space configurations into C-space configurations, and an evaluation of how the reachable volumes sampler scales with roadmap size.

2. Related work

In this section, we give an overview of previous methods that are applicable to motion planning systems with constraints. Many early motion planning methods were able to handle problems with spatial constraints by explicitly computing the set of configurations that satisfy the constraints (Latombe, 1991; Reif, 1979). Unfortunately, their running time is exponential with respect to the number of dofs, which makes them unsuitable for problems with more than four or five dimensions.

Sampling-based motion planning includes graph-based methods (e.g. PRM (Kavraki et al., 1996)) and tree-based methods (e.g. RRT (LaValle and Kuffner, 2001)). Although PRM and RRT have been applied to a wide variety of problems, they both have been shown to be poorly suited for problems with spatial constraints (LaValle et al., 1999). The issue is that the probability of randomly sampling a configuration that satisfies the constraints could be very small and in some cases approaches zero.

Table 1 summarizes the capabilities of the various sampling-based methods. Reachable volume sampling is unique in that it is shown to be applicable to problems with internal joint constraints and problems with constraints on multiple joints, whereas most of the existing methods are limited to end effector constraints. None have been explicitly shown to be applicable to such problems. Reachable volumes are also capable of handling high dof problems, closed chains, and tree-like robots. Unlike other methods, reachable volumes can also handle problems with prismatic and spherical joints and combinations of different types of joints while many existing methods are limited to problems with planar articulated joints. Reachable volumes also has an advantage over RRT-based methods in that it is applicable to multi-query problems. A detailed comparison of these methods follows.

2.1. Adaptations of PRM and RRT methods

PRM and RRT methods have been adapted for use in spatially constrained systems. *Gradient decent methods* push randomly generated configurations onto a constraint surface (LaValle et al., 1999; Yakey et al., 2001). These methods are capable of solving problems with single-loop, articulated joint, closed chains. PRM-MC combines PRM and Monte Carlo methods to generate samples that satisfy closure constraints (Han, 2004). This method can efficiently generate samples for large (100 link) single-loop closed chains. Trinkle and Milgram developed a method that uses C-space analysis for path planning while ignoring self-collisions (Milgram and Trinkle, 2004; Trinkle and Milgram, 2002). They presented results for a set of planar parallel star-shaped manipulators. Alternative task-space and configuration-space exploration (ATACE) for path planning with constrained manipulators uses a randomized gradient decent method for constrained manipulators. Yao and Gupta (2005) presented results for a 9-dof manipulator robot with a set of end-effector constraints. Zhang et al. (2013) presented a Monte Carlo method for generating closed chain samples. This method uses analytical inverse kinematics to ensure that the sub-loops of closed chain robots are sampled in an unbiased manner and is shown to be applicable to two-dimensional chains, closed chains, and protein molecules with over 200 dofs.

There have been a number of RRT-based methods proposed for solving problems with constraints. DDRRT (Yershova et al., 2005; Yershova and LaValle, 2009) reduces the domain for generating samples in highly constrained regions to reduce sampling in directions where no progress is being made. This method has been shown to be applicable to highly constrained problems and to be capable of solving problems with as many as 18 dofs. Atlas-RRT (Jaillet and Porta, 2011) simultaneously builds an RRT and constructs an atlas, the set of charts that locally parametrizes constraint manifolds. The atlas is used to generate samples along the constraint manifolds, which are added to the RRT, whereas the RRT is used to guide the direction in which the atlas is expanded. Tangent bundle RRTs (Suh et al., 2011) construct an RRT along a set of tangent bundles that approximate a problem's constraint manifolds. It then projects the nodes along the solution path onto the manifold so that the solutions are confined to the manifold. This method is shown to be able to solve problems with closed chains and chains with end-effector constraints that have as many as 14 dofs. Unfortunately, it would be difficult to adapt these methods to work in a PRM framework. These methods use samples in the RRT to construct an approximation of the constraint manifolds in the environment. These approximations are only accurate near existing samples, which means they can only be used to generate samples that are near to existing samples. This makes them ideal for constructing RRTs but unsuited for constructing PRMs where nodes need to be generated throughout the environment.

Table 1. Comparison of method capabilities

Method	Constraint Types	High-dof Robots	Closed Chains	Tree-like Robots	Joint Types	Multi-query Problems
Basic PRM (Kavraki et al., 1996)	None	Yes	No	Yes	Planar, Spherical, Prismatic	Yes
Inverse Kinematics (Han and Amato, 2001)	End Effector	Yes	Yes	Yes	Planar	Yes
Constrained Dynamics (Garber and Lin, 2003)	End Effector	Yes	Yes	No	Planar	Yes
PRM with Integrated Sampling and Collision Detection (I-CD, see Algorithm 16)	None	Yes	No	Yes	Planar, Spherical, Prismatic	Yes
DDRRT (Yershova et al., 2005; Yershova and LaValle, 2009),	Only End Effector Shown-	Not shown	Yes	Not shown	Only Planar	No
Atlas RRT (Jaillet and Porta, 2011),	Only End Effector Shown	Not shown	Yes	Not shown	Only Planar	No
Tangent Bundle RRT (Suh et al., 2011)	Only End Effector Shown				Shown	
CCD (Wang and Chen, 1991)	End Effector	No	Yes	No	Planar, Spherical	Yes
CHOMP (Zucker et al., 2013)	Hard and Soft	Yes	Not Shown	Yes	Planar, Spherical, Prismatic, Combinations	No
Reachable Distances (Tang et al., 2010)	End Effector	Yes	Yes	Not shown	Planar, Prismatic	Yes
Reachable Volumes (this paper)	**End Effector, Internal Joints, Multiple Joints**	**Yes**	**Yes**	**Yes**	**Planar, Spherical, Prismatic, Combinations**	**Yes**

2.1.1. Kinematics-based samplers. An alternative approach is to use inverse kinematics to produce constraint-satisfying samples. Kinematics-based PRM utilizes a two-step process (Han and Amato, 2001). First it uses a combination of kinematics and random sampling to generate a roadmap with constraint-satisfying samples and connections that are free of internal collisions. Then it populates the environment with copies of this roadmap, keeping portions that do not collide with obstacles, and connects similar configurations from the different copies of the roadmap using a rigid-body local planner. Cortés et al. developed a sampling method for closed chain linkages with kinematic constraints (Cortés and Siméon, 2005; Cortés et al., 2002). This method is shown to be faster than previous kinematics-based sampling methods. Kinematics-based methods have been extended to large linkages (Xie and Amato, 2004) and multiple loops (Cortés and Siméon, 2003).

Inverse kinematic methods for 3D, 5D, and 6D end-effector constraints are shown to efficiently generate samples for chains with as many as 1,000 links (Han and Rudolph, 2007b,a). They also present the concept of deformation space (D-space) that finds the valid internal motions of a robot, ignoring rigid-body motion. D-space is

analogous to C-space, however it only includes the internal dof of a robot.

It has been shown that for any planar polygonal loop there exist two special configurations such that any connectable pair of configurations can be connected by a sequence of straight line paths through them (Han et al., 2006, 2008). This method has been extended to produce paths guaranteed to be self-collision free (Han et al., 2012). They show that any two convex configurations of a closed chain can be connected by a path comprised of two straight line segments consisting only of convex configurations.

Although inverse kinematics-based methods have had a great deal of success, they also have a number of major limitations. Most of these methods assume a planar robot with 1D planar joints. None of these methods can handle problems with prismatic joints or combinations of different joint types. In addition, these methods are only applicable to end effector constraints; they cannot handle problems with constraints on internal joints or constraints on multiple joints.

2.1.2. Optimization methods. Another approach is to iteratively optimize samples or paths until they satisfy a problems constraints. Cyclic coordinate descent (CCD) (Wang and Chen, 1991) moves the end effector of a robot to a specified end-effector placement by iteratively cycling through the robot's coordinates and adjusting them so that the end effector converges to the goal placement. CCD can also be used to generate closed-chain samples or samples that satisfy a specified end-effector constraint for chains with as many as 7 dofs.

CHOMP (Zucker et al., 2013) uses gradient-based techniques to improve paths by optimizing a function which balances obstacle avoidance and path smoothness. This method can be used to generate paths which satisfy *hard* constraints and optimize adherence to *soft* constraints.

2.1.3. Enforcing constraints during sampling. Another approach is to explicitly enforce constraints while sampling. Han et al. (2006) solved closed-chain problems by transforming them into a system of linear inequalities. Extensions are capable of solving closed-chain problems with multiple loops (Han and Rudolph, 2009). This method is able to handle problems with thousands of links or thousands of loops. Constrained dynamics enforce constraints such as joint connectivity, spatial relationships, and obstacle avoidance for manipulators up to 6 dofs (Garber and Lin, 2003). Other planners require the end effector to traverse a predefined trajectory by generating samples that satisfy the end effector constraints given by the trajectory (Oriolo and Mongillo, 2005; Oriolo et al., 2002). Han et al. (2009) developed a method for generating samples with self-contact, i.e., configurations occurring on the border of C-free and the regions of C-obstacle that denote self-collisions. Although this method uses revolute joints, it requires that loops are planar.

The *reachability grid* is a voxel-based representation that consists of a grid of workspace in which each grid cell is denoted by the minimum time required to reach that cell (Anderson-Sprecher and Simmons, 2012). They show that it is possible to produce accurate reachability grids in real time and that the errors in their estimates are almost always biased towards optimistic ones.

2.2. Reachable workspace and reachable distance.

Reachable workspace (Craig, 1989) is the volume of workspace that can be reached by the center point of the end effector of a fixed-base manipulator. It differs from reachable volumes in that it is only defined for serial linkages and it does not take into consideration a problem's constraints. Moreover, reachable workspace is only defined for end effectors so it cannot be used to generate samples in the same manner as reachable volumes.

The reachable distance of an articulated linkage is the range of distances that its end effector can reach with respect to its base (Tang et al., 2010). Reachable distance is computed by recursively computing the reachable distances of subsets of the linkage. This method efficiently produces samples for linkages, single- and multiple-loop closed chains, and constrained motion planning problems such as writing an on object's surface. Reachable volumes extends this method to handle other joint types including spherical and prismatic joints by computing the *volume* that an end effector (and its subsets) can reach instead of the *range* it can reach. Reachable volumes generalizes the concept of reachable distances for non-planar robots that include 2D spherical joints.

3. Problem formulation: constrained motion planning with linkages

In this section, we describe the types of linkage systems studied in this work and state the motion planning problem with linkages.

3.1. Robot types

We study linkage systems with planar, spherical, and prismatic joints and combinations thereof. These systems consist of a set of links connected to each other by joints. These links can form a chain, in which every joint connects only two links (Figure 1(a)), or a tree, in which some of the joints will connect more than just two links (Figure 1(b)). Closed-chain robots are linkages in which chains of links may form one or many loops (Figure 1(c,d)).

Robot links are assumed to be rigid bodies connected at the ends by joints. These joints may be planar, spherical or prismatic. Planar joints are 1-dof articulated joints. They are represented by a single value denoting the angle of the

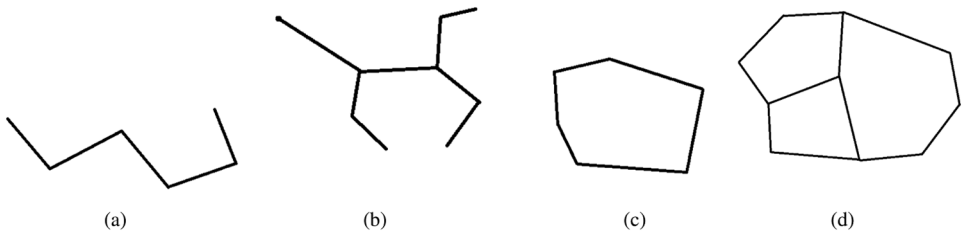

Fig. 1. Examples of linkage systems: (a) an open chain, (b) a tree-like linkage, (c) a closed chain, and (d) a multiple-loop closed chain.

joint (see Figure 2(a)). Linkages connected by adjacent planar joints are coplanar. For chains composed of only planar joints, the entire chain will be coplanar. Spherical joints are 2-dof joints in which any possible angle between adjacent links is valid. They are represented using polar coordinates with an inclination θ and a rotation ϕ (see Figure 2(b)). Prismatic joints are 1-dof linear sliding joints that are represented by a single value d denoting the length by which the joint is extended (see Figure 2(c)).

3.2. Configurations and C-space

A *configuration* is a representation of the position, orientation, and deformation of a robot that consists of a numeric value for each translational, rotational, and joint-angle dof of the robot. *Valid* configurations must satisfy problem or application specific validity constraints. In most applications configurations are considered valid if they do not collide with any obstacles in the environment, however in some applications, such as protein folding and deformable objects, validity is determined by a configuration's physical feasibility. For the sake of simplicity, we will use the terms *valid* and *collision-free* interchangeably.

The set of all possible configurations of a robot, valid or not, forms the robot's *configuration space* (of *C-space*) (Lozano-Pérez and Wesley, 1979). Although it is not feasible in general to explicitly compute which portions of C-space are valid and which portions are not (Reif, 1979), it is efficient to determine whether or not a single configuration is valid, e.g., by performing a collision detection test in the robot's workspace.

3.3. Constrained motion planning with linkages

The objective of the motion planning problem is to locate a valid set of motions (or path) between a start and a goal configuration. For linkage robots, paths consist of deformations or changes in the relative position of the links due to altering the angles of the joints for planar and spherical joints or due to changes in the length of the link for prismatic joints. For free-base linkages, paths also include translational and rotational motions. A path is valid if none of the links collide with each other (self-collision) or with any obstacles present in the environment.

A constrained motion planning problem is defined as a motion planning problem in which a set of constraints $\mathcal{S}$ are applied to some or all of the joints of the robot. As an example, a problem could require that one of the end effectors maintain contact with a surface, or that two of the joints maintain contact with each other so that they form a closed chain. Solutions to constrained motion planning problems must satisfy the constraints in $\mathcal{S}$ along with any other validity conditions associated with the problem.

3.4. Constraints

We define a constraint S_j to be a subset of space in which joint j must be located (see Figure 3). In much of the previous work, constraints were assumed to be placed only on the end effectors of a linkage. Our work is unique in that we allow constraints to be placed on any of the joints or indeed, any point on the robot. Multiple constraints can be applied to the same joint by constructing a single constraint that is their intersection. For example, to apply the constraints $s_{j1}, \ldots, s_{jk}$ to the joint j, you would apply the constraint $S_j = s_{j1} \cap \cdots \cap s_{jk}$. This allows application of both position and workspace constraints to the same joint, as well as to create complex constraints that are the intersection of many constraints. It also implies that $|S|$ equals the number of internal joints plus the number of end effectors. Note that constraints may be absolute (e.g., the end effector must contact an object surface) or relative to another point on the robot (e.g., the end effector must contact the base in a single-loop closed chain).

Joint position constraints can be used to model a wide variety of constrained systems. For example, one can require that a humanoid robot such as a PR2 (see http://www.willowgarage.com) remain upright by constraining its torso to be above its base or require that an object such as a bucket of water remain upright by constraining the vertices of the object to be parallel to the robot's base. Grasping can be modeled by requiring that the end effectors of the robot be located at specified handle positions.

4. Reachable volumes

In this section, we first define the concept of reachable volumes for unconstrained systems. We then extend this definition to incorporate constraints.

The reachable volume of a joint or end effector is the region of space that it can reach and the reachable volume of

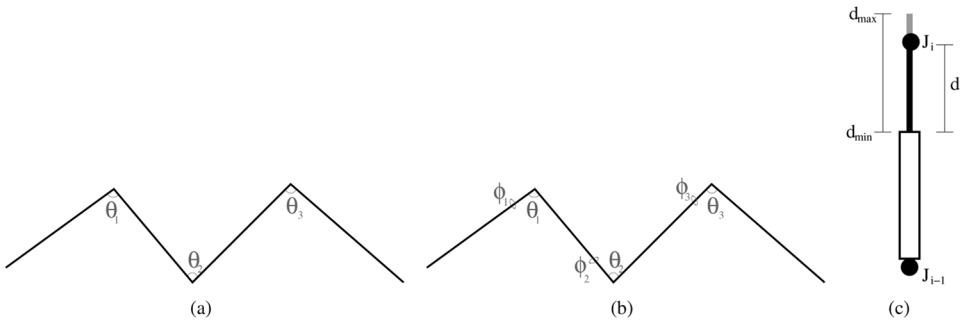

Fig. 2. (a) Planar joints are 1D articulated joints whose motion is confined to a plane. They are represented by a single joint angle coordinate, θ. (b) Spherical joints are represented by an inclination, θ, and a rotation, ϕ. Here, the angles for the first joint are θ_1 and ϕ_1, the angles for the second joint are θ_2 and ϕ_2, and the angles for the third joint are θ_3 and ϕ_3. (c) Prismatic joints are 1D linear sliding joints. They are defined by a distance parameter, d which is between a specified minimum and maximum value (d_{min} and d_{max}).

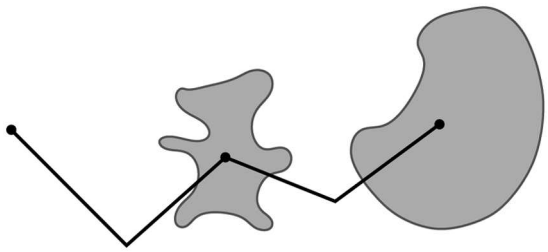

Fig. 3. A chain linkage with constraints. Here, constraints are applied to the end effector and one of the internal joints of the chain as shown by the gray shaded regions.

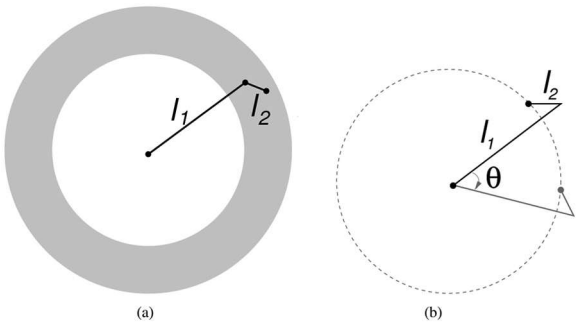

Fig. 4. (a) The reachable volume (gray region) of the end effector of a two-link chain robot, l_1 and l_2 (black). Here l_1 rotates about the point in the center whereas l_2 rotates about the endpoint of l_1. (b) If the end effector (black) can reach a point, then it can reach all other points that are the same distance from the base (dashed gray circle).

a chain is the region of space that its end effector can reach. We use "reachable volume" to refer to a single joint's region of space it can reach and "reachable volumes" to refer to the collection of these regions for a robot where each joint has a single reachable volume associated with it.

In the following, we show that the reachable volume of a chain is equal to the Minkowski sum of the reachable volumes of the links in the chain. This allows us to develop a recursive method for computing the reachable volume of each of the joints in the robot. We show how this approach applies not only to chain linkages, but also to more complex linkages such as trees and closed chains.

4.1. Definitions

We first define a reachable volume space and formally define the concept of reachable volumes. We then show how reachable volumes can be computed and provide visual examples of reachable volumes for a variety of systems.

4.1.1. Reachable volumes space. The *reachable volumes space* (*RV-space*) of a linkage is a 3D space in which the origin is located at one of the joints or end effectors of the robot (referred to as the root). Points in RV-space represent possible locations of the joints and end effectors in the chain with respect to the root. Unlike C-space, RV-space does not include any obstacles; it only encodes the robot's relative dof (e.g., joint angles and lengths) and not the absolute dof (e.g., position and orientation of a mobile base). Thus, RV-space can be used to generate sample configurations that are later tested for validity using a validity checker (e.g. Gottschalk et al., 1996).

We define the reachable volume of a joint or end effector, j, to be the set of points $P \in$ RV-space for which there exists a constraint-satisfying configuration in which j is located at P. We also define the reachable volume of a chain to be the reachable volume of its end effector (see Figure 4(a)).

First note that if the end effector can be placed at one point that is r away from the origin, then it can reach all other points of distance r from the origin by rotating the robot about the origin (see Figure 4(b)). For chains with a single link of length l where the adjacent joint is not prismatic, the reachable set is the set of points that are a distance l from the origin. Thus, the reachable set can be represented by the radii $r_{min} = r_{max} = l$. If the link has an adjacent prismatic joint that ranges between d_{min} and d_{max}, then the reachable set is the set of points represented by the radii $r_{min} = l + d_{min}$ and $r_{max} = l + d_{max}$. Based on this, we observe the following.

Observation 1. *If a point of distance r from the origin is reachable, then all points that are a distance of r from the origin must be reachable.*

Because our definition of RV-space allows the base of a robot to rotate freely about the origin, this observation holds for chains that include planar, spherical, and prismatic joints.

Observation 2. *If a chain can reach a point that is r_1 from the base point and a point that is r_2 from the base point, where $r_1 \leq r_2$, then it can reach all points that are $r_1 \leq r \leq r_2$.*

Both observations hold for chains with planar, spherical, and prismatic joints, however they do not hold for chains with constraints. This is addressed below in Section 4.3.

Based on these observations, there must exist a minimum radius (r_{min}) and maximum radius (r_{max}) such that the reachable volume of a chain is the set of points p whose distance from the origin O is between r_{min} and r_{max}.

$$\texttt{ReachableVolume}(C) = \{p \mid r_{min} \leq \texttt{distance}(p, O) \leq r_{max}\}$$

An RV-space configuration consists of a placement in RV-space for each of the joints and end effectors. A configuration in RV-space is composed of a placement for each joint and end effector in the robot. In an RV-space configuration, the placement of a joint or end effector in RV-space is equal to the difference between the placement of that joint or end effector in workspace and the placement of the root in workspace. An RV-space configuration captures the relative placement of the joints and end effectors. (The placement of the root is handled separately when converting from RV-space to C-space.)

A configuration in RV-space can be transformed into a C-space configuration by computing the joint dof implied by their placement in RV-space and randomly sampling any translations and rotational coordinates of the entire robot. Each joint type (spherical, planar, and prismatic) can be handled in the following way.

- *Spherical joints:* The articulated angle θ can be computed by applying the law of cosines to the triangle formed by the two links that meet at the joint. If (j_{i-1}, j_i) and (j_i, j_{i+1}) are the links that meet at the joint j_i (Figure 5(a)), then the articulated angle for the joint would be $\text{acos}(|\overrightarrow{j_1,j_2}|^2 + |\overrightarrow{j_2,j_3}|^2 - |\overrightarrow{j_1,j_3}|^2 / 2|\overrightarrow{j_1,j_2}||\overrightarrow{j_2,j_3}|)$.
  For the rotational angle φ, the first rotational angle φ_0 is calculated by computing a vector v that is perpendicular to l_0 and l_1 and then computing the angle between this vector and the upward direction (see Figure 5(b)). For all other joints j_i, we compute a vector v that is perpendicular to l_i and l_{i+1} and a vector v' that is perpendicular to l_i and l_{i-1}. φ_i is the angle between v and v' (see Figure 5(c)). Computing each θ and φ value can be done in constant time, which means that an RV-space configuration can be converted to a C-space configuration in linear time with respect to the dofs of the linkage.
- *Planar joints:* The placement of adjacent joints must be coplanar. Thus, planar joints are a subset of spherical joints where φ is always zero. Its single dof θ can be computed in the same way as θ is computed for spherical joints.
- *Prismatic joints:* The distance parameter d can be found by simply computing the distance between the joint and its predecessor.

4.1.2. Relationship between reachable volumes and Minkowski sums. There have been a number of previous applications of Minkowski sums to motion planning. For example, the M-Sum Planner (Lien, 2008) is a hybrid motion planning method that first generates random samples for the angular coordinates of the environment, denoted as C-slices because they represent a slice of C-space in which the angular coordinates are fixed. For each C-slice, the Minkowski sum of the robot and the obstacles in the environment are computed. Then samples are taken along the boundary of the Minkowski sum and samples are connected. Finally, the C-slices are sorted and the nodes in nearby C-slices are connected to form a roadmap. This method generates samples faster than biased samplers and nearly as fast as uniform sampling, solving a set of sample environments faster.

We show that if you attach the base of a chain to the end effector of a second chain, then the reachable volume of the resulting chain is equal to the Minkowski sum of the reachable volumes of the original chains. We also show that the reachable volume of a chain is equivalent to the Minkowski sums of the reachable volumes of the links in that chain.

Lemma 1. *If a chain C can be subdivided into two subchains C_1 and C_2 where C_1 and C_2 share an endpoint, then the reachable volume of C is equal to the Minkowski sum of the reachable volumes of C_1 and C_2.*

Proof. First, observe that a point in the chain's reachable volume can be seen as an offset that is achievable by the chain. For example, if the point (x, y, z) is in the reachable volume, then the corresponding chain can reach a point that is (x, y, z) from the base of the chain (see Figure 6). This is a result of defining the origin to be the first point of the chain.

If C_1 can reach the point (x_1, y_1, z_1) and C_2 can reach the point (x_2, y_2, z_2), then we attach a configuration of C_2 that reaches (x_2, y_2, z_2) to the end of a configuration of C_1 that reaches (x_1, y_1, z_1) to obtain a configuration of C that reaches point $(x_1 + x_2, y_1 + y_2, z_1 + z_2)$. Consequently, if the point (x_1, y_1, z_1) is in the reachable volume of C_1 and the point (x_2, y_2, z_2) is in the reachable volume of C_2, then the point $(x_1 + x_2, y_1 + y_2, z_1 + z_2)$ must be in the reachable volume of C.

Observe that if C can reach a point (x, y, z), then we can take a configuration of C that reaches (x, y, z) and split it

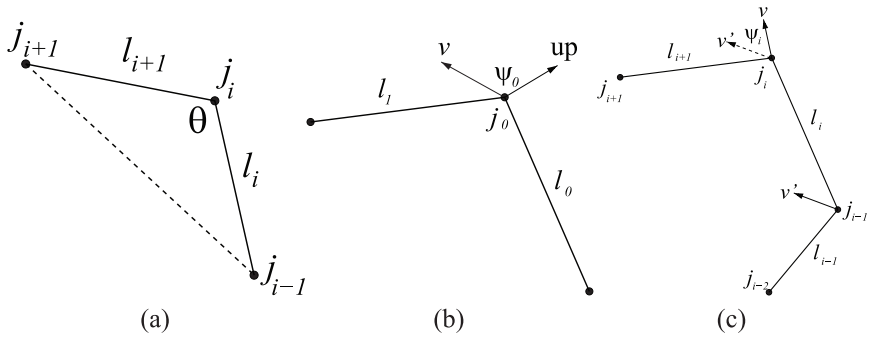

Fig. 5. Computing joint angles. (a) We first compute the joint's articulated angle θ using the law of cosines. (b) The rotational angle φ_0 of the first joint is calculated by computing a vector v that is perpendicular to l_0 and l_1 and then computing the angle between this vector and the upward direction. (c) For all other joints j_i, we compute a vector v that is perpendicular to l_i and l_{i+1} and a vector v' that is perpendicular to l_i and l_{i-1}. Here φ_i is the angle between v and v'.

Fig. 6. The reachable volume of a chain that is composed of two smaller chains C_1 (black) and C_2 (gray) is the Minkowski sum of the reachable volumes of C_1 and C_2.

into configurations of C_1 and C_2 in which the points that C_1 and C_2 reach (in their respective RV-spaces) sum to (x,y,z). We can therefore conclude that for a point (x,y,z) to be in C, there must exist a point (x_1,y_1,z_1) in the reachable volume of C_1 and a point (x_2,y_2,z_2) in the reachable volume C_2 such that $x_1+x_2=x$, $y_1+y_2=y$, and $z_1+z_2=z$.

The reachable volume of C is therefore the following:

$$\texttt{ReachableVolume}(C) = \{(x_1+x_2, y_1+y_2, z_1+z_2) \mid (x_1,y_1,z_1) \in \texttt{ReachableVolume}(C_1) \text{ and } (x_2,y_2,z_2) \in \texttt{ReachableVolume}(C_2)\}$$

This is equivalent to the Minkowski sum of the reachable volumes of C_1 and C_2:

$$\texttt{ReachableVolume}(C) = \texttt{ReachableVolume}(C_1) \oplus \texttt{ReachableVolume}(C_2)$$

□

Corollary 1. *The reachable volume of a chain is the Minkowski sum of the reachable volumes of the links in the chain.*

$$\texttt{ReachableVolume}(C) = \texttt{ReachableVolume}(l_1) \oplus \texttt{ReachableVolume}(l_2) \oplus \cdots \oplus \texttt{ReachableVolume}(l_n)$$

Corollary 1 implies that the reachable volume of a chain can be computed by calculating the Minkowski sum of the reachable volumes of the links in the chain. This computation is shown in Algorithm 1. The reachable volume of a link is a sphere that is centered at the origin and has a radius equal to the length of the link. The reachable volume of a chain is therefore the Minkowski sum of a set of spheres that can easily be computed (see Section 4.2.3). Note that Minkowski sums are commutative (Schneider, 1993), which implies that the order in which the links occur in a chain has no impact on the reachable volume of the chain.

Algorithm 1 `ReachableVolume`(C)

Input: A chain C
Output: The reachable volume of C

```
if C only has one link then
    return A sphere centered at C's base with radius equal to the length of C
Let j be an arbitrary internal joint from C upon which to split C into subchains
Let C_1 ... C_d be each subchain of C that shares joint j (i.e., in a graph representing link connectivity, d is the degree of joint j)
RV = ReachableVolume(C_1)
for i in {2 ... d} do
    RV = RV ⊕ ReachableVolume(C_i)
return RV
```

4.1.3. Reachable volume visualization. We next show a set of examples that illustrate the nature of reachable volumes and demonstrate their capabilities. These include simplistic examples designed to show what reachable volumes will look like for different types of problems as well some complicated examples that show that they are applicable to a wide variety of interesting and useful problems. The reachable volumes for these examples were computed using the method presented in Section 4.4 and displayed using the Vizmo visualization tool (Vargas Estrada et al., 2006). Note that Vizmo displays objects as tetrahedral meshes,

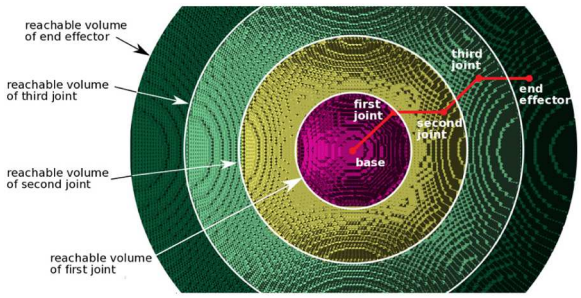

Fig. 7. A cross-section of the reachable volumes of a chain linkage (red) with three spherical joints, four links of equal length, and no constraints. The first joint can reach any point along the inner sphere (pink), the second joint can reach any point inside the second sphere (yellow), the third joint can reach any point inside the third sphere (light green), and the end effector can reach any point inside the outermost sphere (dark green). An example configuration is shown in red.

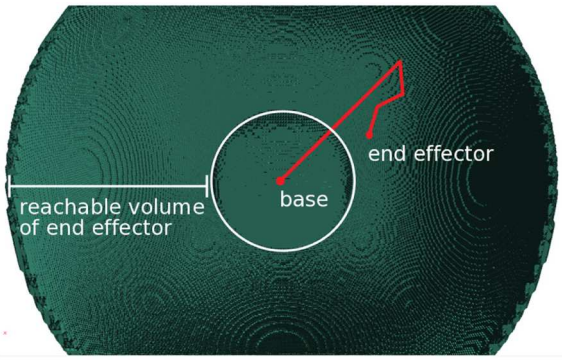

Fig. 8. A cross-section of the reachable volume of the end effector of a chain with one long link and three smaller links. The length of the three smaller links is less than the length of the first long link. The end effector can reach any point between the inner and outer spheres, but it cannot reach the region inside the inner sphere. An example configuration is shown in red.

thus spheres appear voxelized. This is an artifact of the visualization tool.

Figure 7 shows the reachable volumes for each link in a simple four-link chain where each link is the same length and no constraints are present. Each joint in the chain has its own reachable volume sphere. This changes when we change the length of the links in the chain. In Figure 8, we increase the length of one of the links to be longer than the combined length of the other links. The reachable volume of the end effector of this chain is the region between the inner and outer spheres.

The structure of RV-space changes again when a chain is constrained to form a single loop. Figure 9 shows the reachable volumes of our original four-link chain with its end effector constrained to be the same point as the base. The first and third joints can reach any point along the inner sphere (green) whereas the second joint can reach any point inside of the outermost sphere (blue). Two possible configurations are shown (red and yellow).

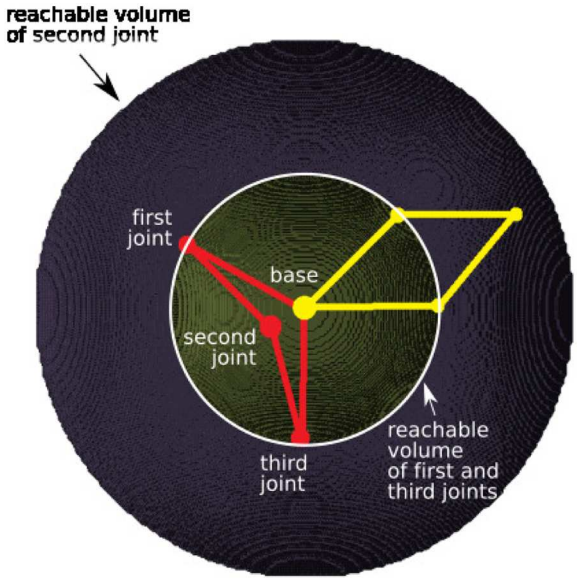

Fig. 9. A cross-section of the reachable volumes of a four-link closed chain with spherical joints. The first and third joints can reach any point along the inner sphere (green) whereas the second joint can reach any point inside the outermost sphere (blue). Example configurations are shown in red and yellow.

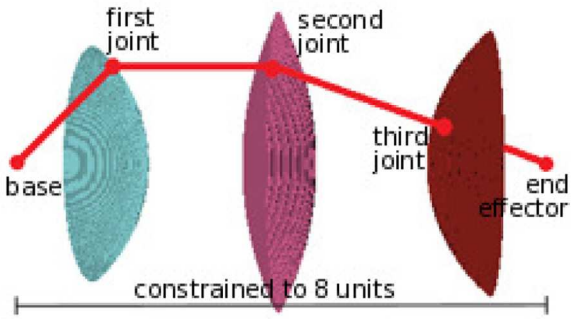

Fig. 10. The reachable volumes of a four-link chain with spherical joints of length 9 where the end effector is constrained to a point 8 units away from the base. The first joint must be located in the left-most shell-like region (blue), the second joint must be located in the middle region (pink), and the third joint must be located in the right-most shell-like region (red). An example configuration is shown in red.

Figure 10 shows the effect of constraining the end effector of the chain. Here the end effector of the chain is constrained to be at a point 8 units away from the base (with the total length of the chain being 9 units). To reach this constraint, the first joint must be located on the left-most shell-like region (blue), the second joint must be located within the center region (pink), and the third joint must be located along the right shell-like region (red).

Figure 11(a) and (b) show reachable volumes of a 16-dof fixed-base grasper with spherical joints in an environment with a set of cubic objects. The wrist denotes the branching point with degree three. Figure 11(a) shows the reachable

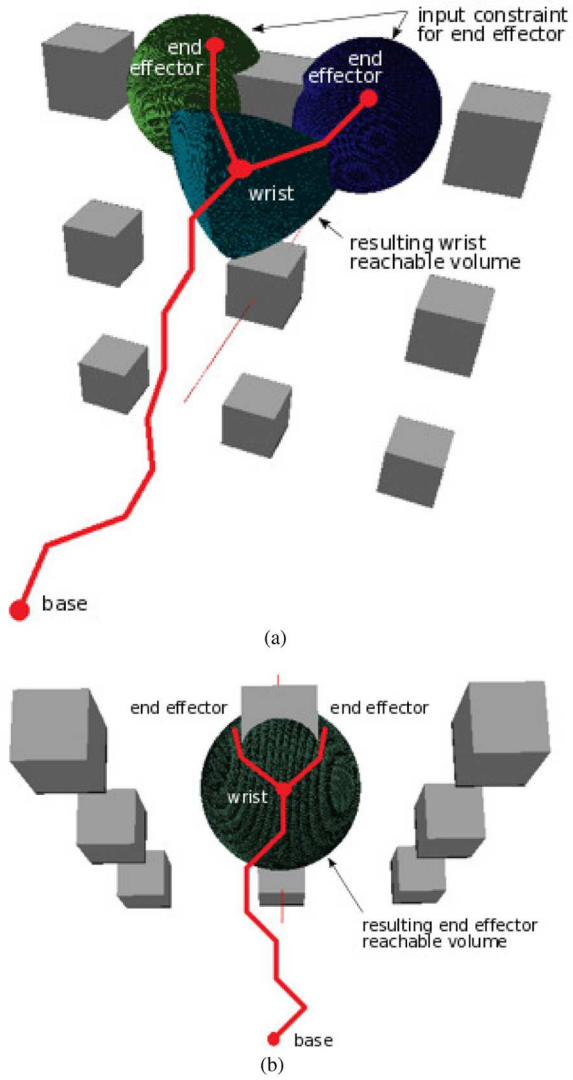

Fig. 11. (a) The reachable volumes of a 16-dof fixed-base grasper with spherical joints is affected by constraints placed either on the end effectors or on the wrist. The reachable volume of the wrist (teal) given constraints on the end effectors to grasp a cubic object (blue and green). (b) The reachable volume of the end effectors (teal) when the wrist is constrained to a specific point. Note that in (b) the end-effector reachable volumes are identical so only one is shown. Example configurations are shown in red.

volume of the wrist when the end effectors are constrained to spherical regions on either side of the object, whereas Figure 11(b) shows the reachable volumes of the end effectors when the wrist is constrained. Note that when the wrist is constrained in such a way, the end effectors have the same reachable volume so only one is shown.

Figure 12 displays the reachable volumes of a WAM robot (see http://www.barrett.com) with 15 dofs and a combination of spherical and planar joints whose end effectors are constrained to grasp a spherical object. To reach the object, the elbow joint must occupy the rightmost region, the second arm joint must be located in the middle region,

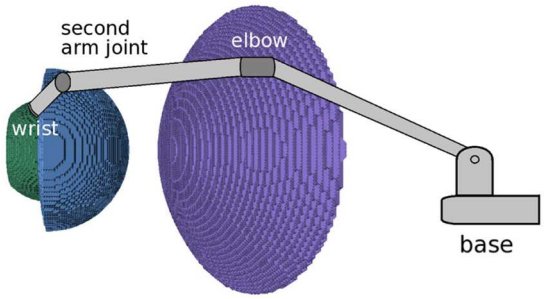

Fig. 12. The reachable volumes of a WAM robot (see http://www.barrett.com) grasping a spherical object. This robot has 15 dofs and includes both spherical and planar joints. To reach the object, the elbow joint must be located in the purple region (right), the second arm joint must be located in the light blue region (center), and the wrist of the grasper must be located in the green region (left). The object being grasped, the knuckle joints, the reachable volumes of the knuckle joints are not visible because they are contained in the reachable volume of the wrist. An example configuration is shown in gray.

and the wrist must be within the left region. The reachable volumes of the knuckles (hidden) are inside this reachable volume.

4.2. Reachable volumes for complex linkages

We next discuss how to compute reachable volumes for complex linkages such as tree-like robots and closed chains. We compute the reachable volumes of these robots by decomposing them into linear chains, computing the reachable volumes of each resulting chain, and then merging them to form the reachable volumes of the robot.

4.2.1. Reachable volumes of tree-like linkages. As with chains, we define the RV-space of a tree-like robot to be a space where the origin is fixed at one of the joints (as described in Section 4.1.1). We then define the reachable volume of each of the tree's end effectors to be the set of points in RV-space that the end effector can reach. Observe that the reachable volume of an end effector in a tree-like linkage is the same as the reachable volume of the chain of links that connects the end effector to the joint located at the origin. The reachable volume of this chain can be computed using the method described in Sections 4.1.1 and 4.1.2. Hence, to compute the reachable volume of a tree-like robot, we can compute the reachable volumes of each of the end effectors of the linkage by computing the reachable volume of the chain that connects it to the joint at the origin (see Figure 13). Note, we can decompose the tree-like linkage into simple linear linkages by first selecting a root joint and then finding the shortest path along the linkage from all remaining leaves to the root.

4.2.2. Reachable volumes of closed chains. As with other robots, we define the RV-space of a closed chain to be a

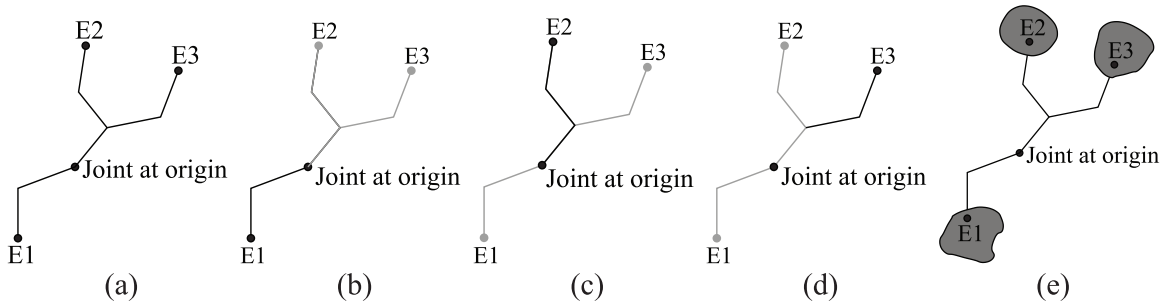

Fig. 13. For a tree robot with three end effectors (E1, E2, and E3), (a) we compute the reachable volumes of each end effector by computing the reachable volumes of the chain connecting it to the origin joint (b–d). This results in the reachable volumes of the end effectors of the linkage (e).

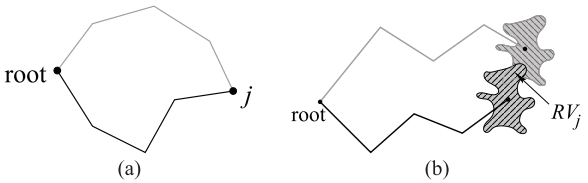

Fig. 14. Generating a closed-chain configuration. (a) Two open chains connect joint j to the root. (b) The reachable volume of j is intersection of the reachable volumes of the adjoining open chains.

space where the origin is fixed at one of the joints. We define the reachable volume of a joint to be the region of RV-space that it can reach. In a single-loop closed chain, each joint is connected to the root by two chains (Figure 14(a)) and the reachable volume of the joint is equal to the intersection of the reachable volume of the chains (Figure 14(b)). Note that once one of the chains is sampled, the reachable volume of the other chain shrinks to a single point located at the first chain's end effector.

For multi-loop closed chains we can compute the reachable volume of a joint by computing the intersection of the reachable volumes of the chains connecting it to the root. We can decompose multi-loop closed chains into a set of single-loop chains and linear chains through ear decomposition (Whitney, 1932, 1933).

4.2.3. Complexity of reachable volumes in problems without constraints. In this section, we study the complexity of reachable volumes. We show that computing a reachable volume has an $\mathcal{O}(1)$ complexity in problems without constraints. We show that this enables us to compute the Minkowski sums of two reachable volumes, which combined with the methods presented in Section 5.1 allows us to generate samples in linear time with respect to the number of joints in the robot.

We first observe that the reachable volume of a chain can be represented by a maximum value which represents the farthest distance from the origin that the chain can reach and a minimum distance that represents the closet point to the origin that the end effector can reach (zero if it can reach the origin). For single-link chains, both the minimum and maximum values are equal to the length of the chain. Using this representation, the reachable volume of a chain is the set of points whose distance from the origin is between these minimum and maximum values.

We next observe that for spherical, planar, and (non-offset) prismatic joints, the reachable volume is the set of points between a specified minimum and maximum distance from the origin. These reachable volumes can be represented in constant space by storing the minimum and maximum distances. Consider a reachable volume $R1$ that is represented by the minimum value $R1_{min}$ and the maximum value $R1_{max}$ and a second reachable volume $R2$ that is represented by the minimum value $R2_{min}$ and the maximum value $R2_{max}$. The Minkowski sum of $R1$ and $R2$ can be represented by the minimum value $(R1 \oplus R2)_{min}$ and the maximum value $(R1 \oplus R2)_{max}$, which are computed as follows:

$$(R1 \oplus R2)_{min} = \begin{cases} \max(R1_{min} - R2_{max}, 0) & \text{if } R1_{min} > R2_{min} \\ \max(R2_{min} - R1_{max}, 0) & \text{otherwise} \end{cases}$$

$$(R1 \oplus R2)_{max} = R1_{max} + R2_{max}$$

We observe that the Minkowski sum of $R1$ and $R2$ is also a reachable volume represented by a minimum and a maximum value. Inductively, we can conclude that the Minkowski sums of the reachable volumes of planar, prismatic, and spherical joints will always be regions within a specified minimum and maximum distance from the origin. We also observe that the Minkowski sum of $R1$ and $R2$ can be computed in constant time regardless of how many joints and links are in the chains that correspond to $R1$ and $R2$.

4.3. Reachable volumes for constrained systems

Here we define reachable volumes for linkages with constraints placed on the placements of its joints and show how to compute them. We define the constrained reachable volume of a chain to be the portion of reachable volume space that the end effector can reach without violating the constraints.

Consider a chain C that is composed of the links $L = \{l_1, l_2, \ldots, l_n\}$ and joints $J = \{j_1, j_2, \ldots, j_n\}$. Let this chain

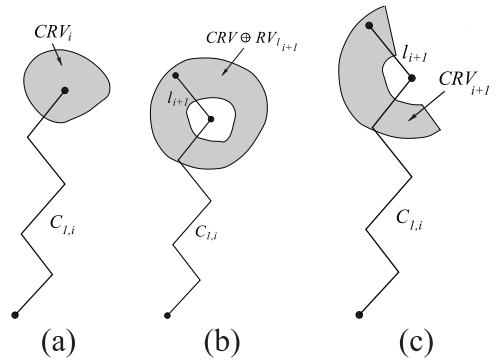

Fig. 15. (a) The constrained reachable volume of $C_{1,i}$ is CRV_i. (b) The region that is reachable by the endpoint of link l_{i+1} is $CRV_i \oplus$ `ReachableVolume`(l_{i+1}). (c) The constrained reachable volume of the chain $C_{1,i+1}$ is therefore $(CRV_i \oplus$ `ReachableVolume`$(l_{i+1})) \cap s_{i+1}$.

have RV-space constraints $S = \{s_1, s_2, \ldots, s_n\}$ placed on its joint placements. We will denote the subset of this chain that includes link l_p through link l_q as $C_{p,q}$. Thus, C may be equivalently referred to as $C_{1,n}$.

As a base case, consider the constrained reachable volume of a chain containing only the first link: $C_{1,1}$. (Here j_1 becomes the end effector for $C_{1,1}$.) The constrained reachable volume of $C_{1,1}$ is `ReachableVolume`$(C_{1,1}) \cap s_1$ which equals `ReachableVolume`$(l_1) \cap s_1$ because $C_{1,1}$ only contains one link. Note that if these do not intersect, no configuration will satisfy the constraint.

We inductively define the constrained reachable volume of the chain $C_{1,i}$ to be CRV_i (see Figure 15(a)). In the chain $C_{1,i}$, the base of link l_{i+1} coincides with the end of link l_i at j_i, and CRV_i is the set of possible locations of j_i. The set of points j_{i+1} can reach is then $CRV_i \oplus$ `ReachableVolume`(l_{i+1}) (see Figure 15(b)), and the set of points that j_{i+1} can reach while satisfying the constraint s_{i+1} then becomes $(CRV_i \oplus$ `ReachableVolume`$(l_{i+1})) \cap s_{i+1}$ (see Figure 15(c)).

By induction, the constrained reachable volume of chain C given by links $L = \{l_1, l_2, \ldots, l_n\}$ and joints $J = \{j_1, j_2, \ldots, j_n\}$ subject to constraints $S = \{s_1, s_2, \ldots, s_n\}$ must be

$$\texttt{ConstrainedReachableVolume}(C, S) = \begin{cases} \texttt{ReachableVolume}(l_1) \cap s_1 & \text{if } |L| = 1 \\ (\texttt{ConstrainedReachableVolume} & \\ (C_{1,|L|-1}, S - s_{|S|}) \oplus \texttt{ReachableVolume} & \\ (l_{|L|})) \cap s_{|L|} & \text{otherwise} \end{cases}$$

Algorithm 2 illustrates the computation.

As with other reachable volumes, the constrained reachable volume does not take into consideration obstacles and it does not exclude configurations with self-collisions. It instead manages constraints on relative joint placement.

Algorithm 2 `ConstrainedReachableVolume`(C, S)

Input: A chain C and a set of joint constraints S
Output: The constrained reachable volume of C subject to S

1: **if** C only has one link **then**
2: **return** `ReachableVolume`$(C) \cap S$
3: Let l be the last link in C
4: Let C_- denote C with l removed.
5: Let s_l be the constraint on the end effector of l.
6: **return** (`ConstrainedReachableVolume`$(C_-, S - s_l) \oplus$`ReachableVolume`$(l)) \cap s_l$

4.3.1. Complexity of reachable volumes in problems with constraints. For constrained problems, the complexity of computing Minkowski sums depends on the geometry of the constraints. Recall that constraints for a joint or end effector are represented as a volume of RV-space in which it can occupy and these volumes may be constructed as intersections of several different individual geometries (see Section 3.4). Thus, these constraint volumes may have arbitrary and possibly complex geometries.

To compute constrained reachable volumes exactly, we need to compute Minkowski sums and intersections on the constraint geometry. For problems where these computations cannot be done efficiently (e.g., constraint geometries are very complex), we first compute the reachable volume of the chain without constraints (using the method presented in Section 4.1.2). Then we separately compute the Minkowski sum of each constraint and the reachable volume of the portion of the chain after the joint where the constraint is applied.

The result is a set of objects whose intersection is the reachable volume of the chain. Minkowski sum operations are commutative, so we can compute `ReachableVolume`$(C_{1,|J|})$ first. Because this is the reachable volume of an unconstrained chain, it is defined by two concentric spheres as before. The Minkowski sum of $s_{|J|}$ and `ReachableVolume`$(C_{1,|J|})$ is the Minkowski sum of $s_{|J|}$ and the area between concentric circles, which can be computed in time proportional to the complexity of $s_{|J|}$. Computing constrained reachable volumes this way requires time $\mathcal{O}(\sum_{s_i \in S} C(s_i))$ where $C(s_i)$ is the complexity of constraint s_i. Samples can therefore be generated in $\mathcal{O}(|J| \sum_{s_i \in S} C(s_i))$ time where $|J|$ is the number of joints. (Recall that the complexity of reachable volume sampling is linear in the complexity of the reachable volumes.)

4.4. Computing the constrained reachable volumes of all joints

In Section 4.3, we presented a method for computing the constrained reachable volume of a single joint (Algorithm 2). This is ideal for an application such as sampling where only the constrained reachable volume of a single joint at any given time needs to be known (i.e.,

Algorithm 3 `ConstrainedReachableVolumeForAllJoints`(R, S)

Input: A robot R represented by a graph $G(J, L)$ that contains no cycles and constraints S on its non-root joints $J \setminus j_0$

Output: The constrained reachable volumes of all joints in R subject to S

1: Let j_0 be the root of R
2: $CRV_{j_0} = \{(0, 0, 0)\}$
3: **for** $i = 1$ to $|J|$ **do**
4: $\quad CRV_{j_i} = s_i$
5: Initialize *changed* to True
6: **while** *changed* **do**
7: $\quad$ *changed* = False
8: $\quad$ **for** $j_i \in J \setminus j_0$ **do**
9: $\quad\quad CRV'_{j_i} = (\bigcap_{j_{neigh} \in Neighbors(j_i)} CRV_{j_{neigh}} \oplus \texttt{ReachableVolume}(l_{j_i, j_{neigh}})) \cap s_i$ where $l_{j_i, j_{neigh}}$ is the link connecting joints j_i and j_{neigh}
10: $\quad\quad$ **if** $CRV_{j_i} \neq CRV'_{j_i}$ **then**
11: $\quad\quad\quad$ *changed* = True
12: $\quad$ **if** *changed* **then**
13: $\quad\quad$ **for** $j_i \in J \setminus j_0$ **do**
14: $\quad\quad\quad CRV_{j_i} = CRV'_{j_i}$
15: **return** $CRV_{j_1}, CRV_{j_2}, \ldots, CRV_{j_{|J|}}$

the joint being sampled). However, for applications such as robot design, control, and modeling, the constrained reachable volume of all of the joints in the robot may be needed. The method presented in Section 4.3 could be applied to every joint in the robot, but that would be very inefficient. In this section, we present an efficient method for computing the constrained reachable volume of all of the joints in the robot. The running time of this method is $\mathcal{O}(|J|\texttt{diameter}(R))$ for unconstrained systems and $\mathcal{O}(|J|\texttt{diameter}(R)\sum_{s_i \in S} C(s_i))$ for constrained systems where $|J|$ is the number of joints in the robot and $\texttt{diameter}(R)$ is the diameter of the robot when expressed as a graph (i.e., joints and end effectors are vertices and links are edges).

Algorithm 3 initializes the constrained reachable volume of each joint to be the constraints for that joint. It then iteratively updates the constrained reachable volume of each joint to be the intersection of its constrained reachable volume and the set of points it can occupy given the constrained reachable volumes of its neighbors. Note that the set of points a joint can occupy given the constrained reachable volumes of its neighbors is equal to the Minkowski sum of the constrained reachable volume of the neighbor and the constrained reachable volume of the link connecting the joint to the neighbor (see Section 4). If the constrained reachable volumes of all joints are not changed over an iteration, the algorithm stops.

To prove the correctness of Algorithm 3, we present the following lemmas.

Lemma 2. *For all $j_i \in J$, the reachable volume of j_i will always be a subset of CRV_{j_i} during all iterations of Algorithm 3.*

Proof. We show this using induction. As a base case, CRV_{j_0} is initialized to $\{(0, 0, 0)\}$ because the root j_0 is at the origin in RV-space by definition. For all other joints j_i, CRV_{j_i} is initialized to the joint's constraints s_i. Because a joint cannot be located outside of its constraint, the constrained reachable volume of j_i must be contained in CRV_{j_i}.

Assume that for all j_i the constrained reachable volume of j_i during iteration k is a subset of CRV_{j_i}. During iteration $k + 1$, this algorithm sets

$$CRV'_{j_i} = \left(\bigcap_{j_{neigh} \in Neighbors(j_i)} CRV_{j_{neigh}} \oplus \texttt{ReachableVolume}(l_{j_i, j_{neigh}}) \right) \cap s_i$$

and then updates CRV_{j_i} to be CRV'_{j_i} if they are different for all joints j_i.

By our inductive assumption, $CRV_{j_{neigh}}$ must include the constrained reachable volume of j_{neigh} for all $j_{neigh} \in Neighbors(j_i)$. The term

$$\left(\bigcap_{j_{neigh} \in Neighbors(j_i)} CRV_{j_{neigh}} \oplus \texttt{ReachableVolume}(l_{j_i, j_{neigh}}) \right) \cap s_i$$

must therefore include the reachable set of j_i. By induction, CRV_{j_i} must always include the constrained reachable volume of j_i. □

Lemma 3. *Let $R^k_{j_i} \subseteq R$ be the subset of the robot R that is within k hops of j_i. After the kth iteration, CRV_{j_i} will be a subset of the constrained reachable volume of j_i that allows satisfaction of all the constraints associated with joints in $R^k_{j_i}$.*

Proof. We show this using induction. As a base case, CRV_{j_i} is set to s_i, which is the constrained reachable volume of j_i in $R^0_{j_i}$ (which consists only of the joint j). Thus, all constraints associated with $R^0_{j_i}$ are satisfied.

Assume that for all $j_i \in J$, CRV_{j_i} is a subset of the constrained reachable volume of j_i that allows satisfaction of all the constraints associated with joints in $R^k_{j_i}$. During iteration $k + 1$, Algorithm 3 sets

$$CRV'_{j_i} = (\bigcap_{j_{neigh} \in Neighbors(j_i)} CRV_{j_{neigh}} \oplus \texttt{ReachableVolume}(l_{j_i, j_{neigh}})) \cap s_i$$

for all $j_i \in J$. We first note that the graph $R^{k+1}_{j_i}$ must be a tree because it is a connected subset of R which had by definition no cycles. As a convention, we define j_i to be the

root of $R_{j_i}^{k+1}$. Let j_{neigh} be an arbitrary neighbor of j_i in $R_{j_i}^{k+1}$. j_{neigh} and all of its descendants are in $R_{j_i}^{k}$, so by our inductive assumption CRV'_{j_i} must be a subset of the region reachable by j_{neigh} under the constraints of j_{neigh} and its descendants in $R_{j_i}^{k+1}$. $CRV_{j_{neigh}} \oplus \texttt{ReachableVolume}(l_{j_i,j_{neigh}})$ must therefore be a subset of the region that j_{neigh} can reach while satisfying the constraints of j_{neigh} and its descendants.

$$\left(\bigcap_{j_{neigh} \in Neighbors(j_i)} CRV_{j_{neigh}} \oplus \texttt{ReachableVolume}(l_{j_i,j_{neigh}}) \right) \cap s_i$$

must therefore be a subset of the region that j_i can occupy while satisfying its own constraint and the constraints of all other joints in $R_{j_i}^{k+1}$. □

Corollary 2. *After at most* $\texttt{diameter}(R)+1$ *iterations, Algorithm 3 will return the constrained reachable volume of all joints.*

Proof. First we observe that the constrained reachable volume of a joint j_i cannot contain any points that are not in s_i, otherwise the constraint s_i would be violated. We also observe that if two joints j_i and j_{neigh} are connected by an edge $e(j_i, j_{neigh})$, then every point p in the constrained reachable volume of j_i must be $|e|$ away from a point in the constrained reachable volume of j_{neigh}, otherwise it would be impossible to place j_i at p without placing j_{neigh} outside of this constrained reachable volume. From these observations we conclude that if at iteration k, $CRV_{j_0}, CRV_{j_1}, \ldots, CRV_{j_{|J|}}$ is equal to the constrained reachable volumes of R, then for all $j_i \in J$,

$$\left(\bigcap_{j_{neigh} \in Neighbors(j_i)} CRV_{j_{neigh}} \oplus \texttt{ReachableVolume}(l_{j_i,j_{neigh}}) \right) \cap s_i$$

This means that $CRV_{j_i} = CRV'_{j_i}$ for all j_i and that the algorithm will return CRV'_{j_i} (which is equal to CRV_{j_i} on the next iteration) for all $j_i \in J$.

For all $j_i \in J$, $R_{j_i}^{d} = R$ where $d = \texttt{diameter}(R)$, so by Lemma 2 $RV_{j_i}^{d}$ must be a subset of the constrained reachable volume of j_i in R. In Lemma 2 we showed that for all iterations k the constrained reachable volume of j_i will be a subset of CRV_{j_i}, which means that CRV'_{j_i} on the $\texttt{diameter}(R)$th iteration must be equivalent to the constrained reachable volume of j_i. Because this contains the constrained reachable volume of all the joints in R, we know that Algorithm 3 will return this constrained reachable volume on the next iteration.

Now consider the case where Algorithm 3 returns on some arbitrary iteration $k \leq \texttt{diameter}(R)+1$. For it to return, CRV_{j_i} on the prior iteration must be equal to CRV_{j_i}. If this algorithm were to continue running, then inductively all CRV_{j_i} for iterations $k' > k$ must be equal to CRV_{j_i} on the kth iteration. CRV_{j_i} on this iteration must therefore be equal to CRV_{j_i} on the $(\texttt{diameter}(R)+1)$th iteration, which is the constrained reachable volumes of R. Each iteration requires $\mathcal{O}(|J| \sum_{s_j \in S} C(s_j))$. □

Algorithm 4 `SampleRVChain`(C)

Input: A chain C with n joints
Output: A randomly sampled configuration of C that specifies a value for each dof

1: Let $P = \{p_1, p_2, \ldots, p_n\}$ store joint placements as they are sampled, initialize each placement to $\emptyset$
2: $RV_C = \texttt{ReachableVolume}(C)$
3: Randomly sample p_n from RV_C for the placement of C's end effector
4: $P = \texttt{SampleRV}(C, P)$
5: Randomly sample the translational and rotational dofs of C's base d_{base}
6: **return** $\texttt{ConvertToCSpace}(C, P, d_{base})$ ▷ as described in Section 4.1.1

5. Sampling-based motion planning with reachable volumes

In this section, we show how sampling-based motion planning can be used with RV-space. We first present RV-space versions of the primitives such as sampling, local planning, and distance computation required by sampling-based motion planning. We then show how they can be used in planning.

5.1. Reachable volume sampling

We describe how reachable volumes can be used to compute configurations for chains, tree-like robots, and closed chains without joint constraints. We then describe how reachable volumes can be used to generate samples for these robot types with constraints.

5.1.1. Generating configurations for chains. Algorithm 4 outlines how reachable volume sampling is done for single chains without constraints. It first computes the reachable volume of the chain's end effector. For problems requiring multiple samples, this computation can be performed once as a prepossessing step because RV-space only depends on the robot, not the environment/workspace in which it operates. We then randomly select an end-effector placement from this reachable volume. We place the internal joints through a recursive algorithm (Algorithm 5) and sample any translational and rotational dofs of the chain's base. We use the resulting base placement, internal joint placements, and end-effector placement to compute the corresponding C-space configuration as described in Section 4.1.1.

To sample the internal joints, we first select a "splitting" joint to "break" the chain into two sub-pieces (e.g., the joint nearest the chain's midpoint such that the two resulting subpieces have similar numbers of joints). This splitting joint becomes the end effectors for the two subchains. For each subchain, the base is already known (previously sampled). We sample their end effector placement by finding a random point in the intersection of their reachable volumes. Note

Algorithm 5 `SampleRV`(C, P)

Input: A chain C with links $L = \{l_1, l_2, \ldots, l_n\}$ and joint placements $P = \{p_0, p_1, \ldots, p_n\}$ where $p_i = \emptyset$ if it has not yet been sampled

Output: Randomly sampled joint placements P'

1: **if** $|C|$ is 1 **then**
2: **return** P
3: Select the joint j nearest the midpoint of C (i.e., in terms of C's cardinality)
4: Let C_1 contain the links $\{l_1, l_2, \ldots, l_j\}$ and $P_1 = \{p_0, p_1, \ldots, p_j\}$
5: Let C_2 contain the links $\{l_n, l_{n-1}, \ldots, l_{j+1}\}$ and $P_2 = \{p_n, p_{n-1}, \ldots, p_j\}$
6: $RV_{C_1} =$ `ReachableVolume`(C_1)
7: $RV_{C_2} =$ `ReachableVolume`(C_2) translated by $(p_n - p_0)$
8: Randomly sample p_{rand} from $RV_{C_1} \cap RV_{C_2}$
9: Set $p_j \in P_1$ to $(p_{rand} + p_0)$ and $p_j \in P_2$ to $(p_{rand} + p_n)$
10: `SampleRV`(C_1, P_1)
11: `SampleRV`(C_2, P_2)
12: **return** $\{p_0 \in P_1, \ldots, p_j \in P_1, p_{j+1} \in P_2, \ldots, p_n \in P_2\}$

that for the second subchain, we must translate its reachable volume by the distance between the original chain's end effector and base to account for origin shifts in RV-space. We then recursively place the internal joints of each subchain, see Figure 16.

For planar and prismatic joints, we traverse the planar/prismatic joints first. This ensures that subchains containing planar and prismatic joints will be sampled after any adjacent spherical joints. Subchains containing planar and prismatic joints must be coplanar, so after we sample the first joint of a subchain we constrain all other points to be in the plane defined by this point and the endpoints of the subchain. The sample space of the joints of the subchain is therefore the intersection between this plane and the joint's reachable volume.

We next show that the running time of the reachable volume sampler is linear in the number of links for problems without constraints.

Proof. The sampler first computes the reachable volume of the chain by recursively breaking the chain. At the bottom level of this recursion, the sampler computes and returns the reachable volume of a single link, which can be done in constant time and is done once per link. It then computes the Minkowski sums of these reachable volumes, performing a total of $\mathcal{O}(|L|)$ Minkowski sum operations (where $(|L|)$ is the number of links in the robot). In Section 4.2.3 we showed that the complexity of computing the Minkowski sums of two reachable volumes is proportional to the complexity of the individual reachable volumes and that this complexity is $\mathcal{O}(1)$ in problems without constraints. The cost of this step is therefore $\mathcal{O}(|L|)$.

Once the reachable volume of the chain is computed, the algorithm samples each joint in the order that they were subdivided when computing the reachable volume of the chain. Consider an internal joint j that breaks the chain into $\{j_l, \ldots, j\}$ and $\{j, \ldots, j_r\}$. When computing the reachable volume of the chain in the first step of Algorithm 5, we recursively compute the reachable volumes of the chain $\{j_l, \ldots, j\}$ (which we will denote as $RV_{j_l,j}$) and $\{j_r, \ldots, j\}$ (which we will denote as $RV_{j_r,j}$). Note that for presentation, we flip the second subchain so that its end effector is the same joint as the first subchain's end effector. However, reachable volumes without constraints are symmetric so this flipping is not necessary in the implementation. Because $RV_{j_l,j}$ and $RV_{j_r,j}$ were computed while computing the reachable volume of the chain in Algorithm 4, we do not need to compute again them during internal joint sampling. Algorithm 5 samples j by placing j in the intersection of $RV_{j_l,j}$ and $RV_{j_r,j}$ translated by $(j_r - j_l)$ (to account for the chain's base always being located at the RV-space origin). This translation can be done by redefining the base location (which can be done in constant time).

Samples are generated in reachable volume intersections by computing a bounding box or patch around the intersection of $RV_{j_l,j}$ and $RV_{j_r,j}$ (which can be done in constant time), and then repeatedly generating samples and testing if they are in $RV_{j_l,j}$ and $RV_{j_r,j}$. (More details are given in Section 5.1.3.) Testing whether a point is in a reachable volume is equivalent to testing whether the distance between the point and the base of the chain is between the minimum and maximum values for that chain, which can be done in constant time. Because we limit the total number of attempts to be a predefined constant, the total time to sample a joint's placement (or return failure) is $\mathcal{O}(1)$. The total time to sample all the internal joints in the chain is therefore $\mathcal{O}(|L|)$.

In the final step, we convert the sample into a joint angle sample, which can be done in $\mathcal{O}(|L|)$ time as described in Section 4.1.1. The total running time of the reachable volume sampler for problems without constraints is therefore linear with respect to the number of links in the robot. □

5.1.2. Generating configurations for complex linkages. We next discuss how to generate samples for complex linkages such as trees and closed chains. For such linkages we decompose the robot into chains and sample the end effectors of these chains. We then sample the internal joints of these chains in the same manner that we sample the internal joints of open chains.

Generating configurations for tree-like robots: To generate configurations for linkages with branches, we partition the linkage into a set of disjoint chains (see Figure 17) by dividing the tree at any joint with more than two neighbors. We partition the robot and order the subchains such that each subchain only shares its base with the preceding subchains. It may share any of its joints with successive subchain bases, but may not share its non-base joints with

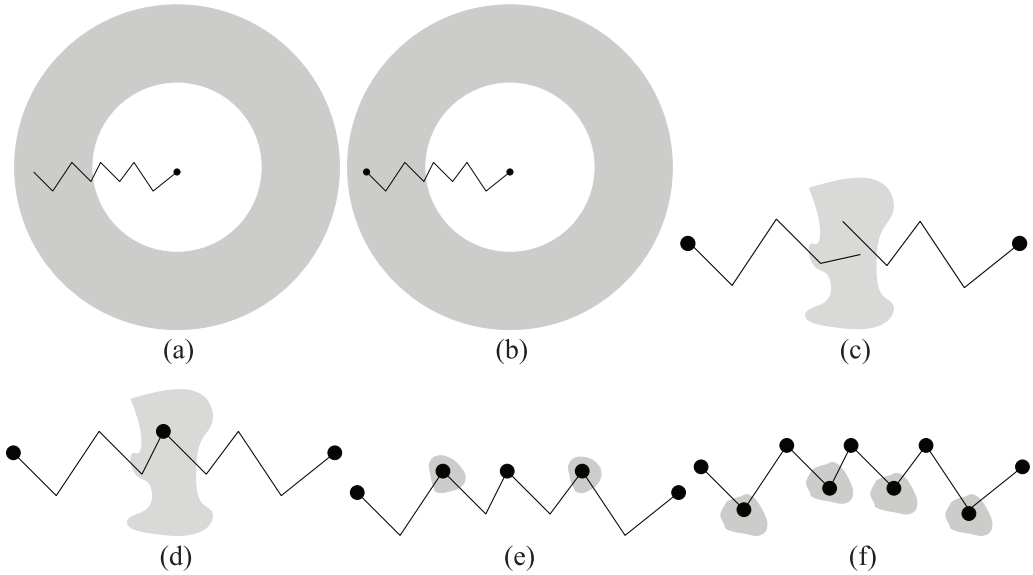

Fig. 16. Generating a configuration for a chain robot. (a) Compute the reachable volume of the chain. (b) Set the placement of the end effector of the chain to be a random point from this volume. (c) Bisect the chain and compute intersection of the reachable volumes of the two pieces. (d) Set the midpoint joint of the bisected chain to be a random point from the intersection of these reachable volumes. (e), (f) Continue until all joints are placed.

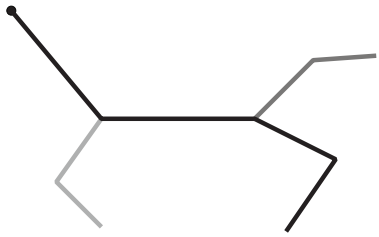

Fig. 17. We partition a tree-like robot into a set of disjoint chains by splitting at every joint with more than two neighbors. This example yields three chains: black, dark gray, and light gray.

preceding subchains. Other than this restriction, the order in which we partition the robot does not affect computation time or the probability distribution of the sample, so we randomly select an ordering that meets these conditions.

We then generate an RV-space configuration for each chain and stitch them together much like was done in Algorithm 5 for both halves of the split chain. We merge the resulting joint placements together, sample the dof for the base, and compute the corresponding C-space configuration. Algorithm 6 outlines this process.

Algorithm 4 samples each decomposed subchain in linear time with respect to the size of the decomposed subchain, so the total time to sample all decomposed subchain is linear in the number of links in the robot. Translating each chain requires us to translate each of the joints in the chain and can be done in linear time with respect to the size of the chain so that the total time required to translate all the chains is linear in the number of joints (or links) in the robot. Converting the sample to a C-space sample can also be done in linear time (as described in Section 4.1.1), so the time required to generate samples for tree-like robots (without constraints) is also $\mathcal{O}(|L|)$.

Algorithm 6 `SampleRVTree`(T)

Input: A tree-like robot T

Output: A randomly sampled configuration of T that specifies a value for each dof

1: Decompose T into disjoint subchains $C_1, C_2, \ldots, C_m$ such that each subchain C_i shares only its base with one or more other subchains C_j for all $j < i$
2: **for** $i = 1$ to m **do**
3: Let $P_i = \{p_1, p_2, \ldots, p_n\}$ store the joint placements for C_i as it is sampled, initialize each placement to $\emptyset$
4: $RV_{C_i} =$ `ReachableVolume`(C_i)
5: **if** p_1 is a shared joint with a chain C_j where $j < i$ **then**
6: Set p_1 to the value of the shared placed joint in C_j
7: Randomly sample p_n from RV_{C_i} for the placement of C_i's end effector
8: $P_i =$ `SampleRV`(C_i, P_i)
9: Merge $P_1, P_2, \ldots, P_m$ into a single list of joint placements P
10: Randomly sample the translational and rotational dofs of C's base d_{base}
11: **return** `ConvertToCSpace`(C, P, d_{base}) ▷ as described in Section 4.1.1

Generating configurations for closed chains: To compute configurations for closed chains, we decompose the closed chain into two open chains. If these two open chains are in configurations that share the same endpoints, they form a closed-chain configuration (see Figure 18(a)). We

Algorithm 7 `SampleRVSingleLoop`(C)

Input: A single-loop closed chain C

Output: A randomly sampled configuration of C that specifies a value for each dof

1: Let j_{base} and j_{ee} be arbitrary joints from C such that $j_{base} \neq j_{ee}$
2: Let C_1 and C_2 be the two chains formed by breaking C at j_{base} and j_{ee}
3: Let $P_1 = \{p_{base}, p_{base+1}, \ldots, p_{ee}\}$ store the joint placements for C_1 as it is sampled and $P_2 = \{p_{ee}, p_{ee+1}, \ldots, p_{base}\}$ store the joint placements for C_2 as it is sampled , initialize each placement to $\emptyset$
4: $RV_{C_1} =$ `ReachableVolume`(C_1)
5: $RV_{C_2} =$ `ReachableVolume`(C_2)
6: Randomly sample p_{ee} from $RV_{C_1} \cap RV_{C_2}$
7: `SampleRV`(C_1, P_1)
8: `SampleRV`(C_2, P_2)
9: $P = P_1 \cap P_2$
10: Randomly sample the translational and rotational dofs of C's base d_{base}
11: **return** `ConvertToCSpace`(C, P, d_{base}) ▷ as described in Section 4.1.1

can ensure this is the case by defining the chains such that their bases are shared and place their end effectors somewhere in the intersection of each chain's reachable volume (see Figure 18(b)). We then sample each chain's internal joints as before to meet this end-effector placement (see Figure 18(c)). Algorithm 7 describes this process.

The reachable volume of each chain is found by computing the Minkowski sums of the links in the chain, which can be done in linear time for problems without constraints (see Section 4.2.3). A sample is then generated in the intersection of these reachable volumes, which can be done in constant time using the methods from Section 5.1.3. Each chain can then be sampled in linear time (as described in Section 5.1.1), so the time required to generate samples for closed chains is also $\mathcal{O}(|L|)$ in problems without constraints.

5.1.3. Sampling in reachable volume intersections. We present a set of methods for computing samples in reachable volume intersections and discuss when each method is applicable. These methods can be used by the algorithms in Sections 5.1.1–5.1.2 for sampling joint placements.

The *intersection* method is applicable to reachable volumes that are the intersection of spheres. This method selects a random point along the circle formed by this intersection. This method is useful for sampling joints where two or more neighbors have already been sampled.

The *bounding patch* method is applicable to reachable volumes that are the intersection of a sphere-like reachable volume and a set of other reachable volumes. This method constructs a patch on the surface of the sphere that encompasses the intersection with the other reachable volumes. It then samples on this patch until it finds a joint that is in all of the other reachable volumes. This method is useful for joints where one neighbor has already been sampled.

The *bounding cube* method constructs an axis-aligned bounding box around the reachable volumes and then samples within this bounding box until it finds a sample that is in all of the reachable volumes. This method is used to sample joints where no neighbors have already been sampled.

The *brute force* method randomly selects points from an arbitrary reachable volume until it locates a point that is in all of the other intersecting reachable volumes. This method can be applied to situations in which none of the other methods are applicable.

Observation 3. *Each of these methods is complete in that they sample over a joint's entire reachable volume. Consequently, they can be used by the reachable volume samplers to provide probabilistically complete sampling. In addition, the complexity of these methods is linear with respect to the number of reachable volumes involved.*

5.1.4. Generating configurations for constrained systems. We next develop a sampler for problems with internal joint constraints as well as for tree-like graspers with constraints on their end effectors. It would be possible to compute samples for constrained problems in the same manner as unconstrained problems, however our proof for the linear time complexity (Section 5.1.1) does not hold for problems with constraints. This proof relies on the symmetry of reachable volumes without constraints to reuse reachable volume computations of the full chain for various subchains. Unfortunately, reachable volumes for problems with constraints are not symmetric. This means that we cannot reuse the reachable volume computations from the full chain for flipped subchains (i.e., the base becomes the end effector and the end effector becomes the base). To use the methods described in Sections 5.1.1 and 5.1.2 for constrained problems, it would be necessary to compute subchain reachable volumes during each level of sampling. Because these methods use a bisecting strategy, they would require $\mathcal{O}(|L| \log(|L|))$ time.

Algorithm 8 shows how to use constrained reachable volumes to compute samples for problems with constraints. It selects the first joint as the base and initializes its placement to the origin. This joint will later be sampled to convert the RV-space configuration into a C-space configuration (see lines 10 and 11). It will use a map M to save the partial constrained reachable volumes as they are computed for future use. It decomposes the chain into subchains, one for each end effector of the original chain. It then computes the partial constrained reachable volume for each of these subchains, storing these for each joint in the subchain in M (see Algorithm 9). It randomly samples the subchain's end effector placement from the resulting reachable volume. It samples the remaining joints in the subchain through Algorithm 10. This algorithm performs a second

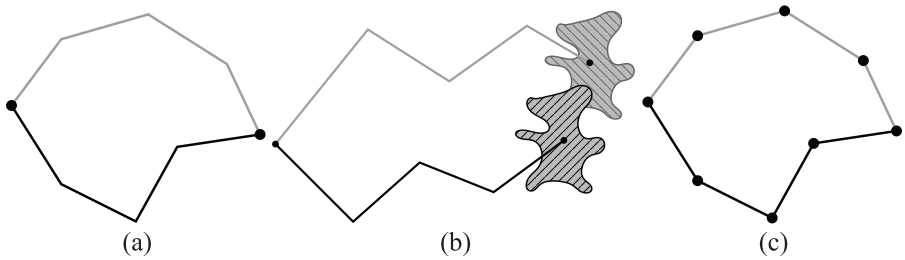

Fig. 18. Generating a closed-chain configuration. (a) Break the closed chain into two open chains. (b) Compute the reachable volumes of the two chains (striped regions). (c) Randomly select a point from the intersection of their reachable volumes and use Algorithm 4 to sample the placements of the internal joints.

Algorithm 8 `SampleCRVChain`(C, S)

Input: A chain (or tree without cycles) C with n joints and a set of constraints S on those joints

Output: A randomly sampled configuration of C that specifies a value for each dof and satisfies S

1: Let $P = \{p_1, p_2, \ldots, p_n\}$ store joint placements as they are sampled, initialize each placement to $\emptyset$
2: Select j_1 as the base and set p_1 to the origin
3: Let M be a map that stores the partial constrained reachable volume for each joint as they are computed for reuse
4: Decompose C into subchains $C_1, C_2, \ldots, C_m$ such that m is the number of C's end effectors and for each C_i, its base is j_1 and its end effector is an end effector of C
5: **for** $i = 1$ to m **do**
6: Let S_i and P_i denote the subsets of S and P corresponding to C_i
7: RV_{C_i} = `PartialConstrainedReachableVolume`(C_i, S_i, P_i, M) ▷ see Algorithm 9
8: Randomly sample p_i from RV_{C_i}
9: P_i = `SamplePartialCRV`(C_i, P_i, M) ▷ see Algorithm 10
10: Randomly sample the translational and rotational dof of C's base d_{base}
11: **return** `ConvertToCSpace`(C, P, d_{base}) ▷ as described in Section 4.1.1

Algorithm 9 `PartialConstrainedReachableVolume`(C, S, P, M)

Input: A chain (or tree without cycles) C with n joints, a set of constraints S on those joints, joint placements P where $p_i = \emptyset$ if it has not yet been sampled, and a map M that stores the partial constrained reachable volume for each joint as they are computed for later use

Output: The constrained reachable volume of C subject to S and P

1: **if** $p_n \neq \emptyset$ **then**
2: Add $(j_n, \{p_n\})$ to M
3: **return** $\{p_n\}$
4: **else**
5: $RV = s_n$
6: **for all** neighbors j' of j_n such that $p' = \emptyset$ **do**
7: Let C' denote the subset of C containing the connected component of j' if j_n were removed
8: Let S' and P' denote the corresponding subsets of S and P
9: RV = $RV \cap$ `PartialConstrainedReachableVolume`(C', S', P', M)
10: Add (j_n, RV) to M
11: **return** RV

depth first traversal of the chain and leverages the reachable volumes already computed and stored in M. Finally, it randomly samples the translational and rotational dof of the base (j_1) and converts the RV-space configuration into a C-space configuration.

When sampling a subchain's internal joints in Algorithm 10, the placement of every joint is set to be a random point in the intersection of its partial constrained reachable volume (stored in M) and its reachable volume given the sampled placement of its parent joint. The joint's j_{n-1} reachable volume given the placement of its parent joint j_n is computed as follows.

- If j_n is a spherical joint, then `ReachableVolume`$(C_{n-1,n})$ is a sphere centered at j_{n-1} with a radius equal to the length of the link between j_{n-1} and j_n.
- If j_n is a planar joint, then `ReachableVolume`$(C_{n-1,n})$ is a circle in the plane of j_{n-1} with a radius equal to the length of the link between j_{n-1} and j_n.
- If j_n is a prismatic joint, then `ReachableVolume`$(C_{n-1,n})$ is the line segment defined by the points d_{min} and d_{max} for j_n (see Section 3.1).

These algorithms perform two depth-first traversals of the robot. During the first traversal, we compute the partial constrained reachable volume of each joint from the partial constrained reachable volumes of its children in the traversal. This requires one Minkowski sum operation and one intersection operation for each link in the robot. During the second traversal, we sample each of the joints of the robot, compute the resulting reachable volume of the link connecting the joint to its parent, compute the intersection of this reachable volume and the partial constrained reachable volume of the joint (computed in the first traversal), and set the placement of the joint to be a random point from this intersection. This method preforms $\mathcal{O}(|L|)$

Algorithm 10 `SamplePartialCRV`(C, P, M)

Input: A chain C with n joints, a set of (partial) joint placements P where p_n has been sampled and $p_i = \emptyset$ denotes a joint that has not yet been sampled, and a map M storing the partial constrained reachable volumes for each joint (as computed by Algorithm 9)

Output: Randomly sampled (partial) joint placements P'

1: **if** $p_{n-1} \neq \emptyset$ **then**
2: **return** P
3: **else**
4: Let C' denote the subchain of C from j_1 to j_{n-1} and S' and P' denote the corresponding subsets of S and P
5: Let C'' denote the subchain (i.e., a single edge) of C from j_n to j_{n-1}
6: $RV' = M[j_{n-1}] \cap (p_n \oplus$ `ReachableVolume`$(C''))$
7: **if** $RV' \neq \emptyset$ **then**
8: Randomly sample p_{n-1} from RV'
9: **return** `SamplePartialCRV`(C', P', M)
10: **else**
11: **return** failure (i.e., no set of joint placements exists that satisfies S and also contains the partial joint placements P)

Algorithm 11 `SampleCRVSingleLoop`(C, S)

Input: A single-loop closed-chain C with n joints and a set of constraints S on those joints

Output: A randomly sampled configuration of C that specifies a value for each dof and satisfies S

1: Let $P = \{p_1, p_2, \ldots, p_n\}$ store joint placements as they are sampled, initialize each placement to $\emptyset$
2: Select j_1 as the base and set p_1 to the origin
3: Let M be a map that stores the partial constrained reachable volume for each joint as they are computed for reuse
4: Let C' denote the subchain of C from j_1 to j_n and S' and P' denote the corresponding subsets of S and P
5: Let C'' denote the subchain (i.e., a single edge) of C from j_n to j_1
6: Set $s_n \in S'$ to $s_n \cap$ `ReachableVolume`(C'')
7: $RV =$ `PartialConstrainedReachableVolume`(C', S', P', M)
8: Randomly sample p_n from RV
9: $P =$ `SamplePartialCRV`(C', P', M)
10: Randomly sample the translational and rotational dofs of C's base d_{base}
11: **return** `ConvertToCSpace`(C, P, d_{base}) ▷ as described in Section 4.1.1

Minkowski sum operations and $\mathcal{O}(|L|)$ intersection operations, so its running time is $\mathcal{O}(|L|)$ in the complexity of these operations.

We next discuss how to use reachable volumes to generate samples for closed chain robots with constraints (Algorithm 11). A single-loop closed-chain robot is any robot of genus two. We first define the base of this robot to be one of the joints along the closed chain of the robot (such that if this joint were removed, the robot would contain no cycles). We denote this as j_1. We then select one of j_1's neighbors on the closed chain. We denote this neighboring joint as j_n. Because the base is at the origin of RV-space by definition, the closed chain can be represented as a linear chain with an additional constraint on j_n that it remain a distance d away from j_1 (i.e., the origin) where d is the length of the link connecting j_n and j_1. This is done by setting j_n's constraint s_n to be the intersection of the original joint constraint and `ReachableVolume`$(C_{n,1})$. Recall that unconstrained joints may be represented as having a constraint equal to the entire workspace. We can now sample the resulting subchain as before in Algorithm 8.

This algorithm performs two depth-first traversals on each subchain of the robot. As with Algorithm 8, this method preforms $\mathcal{O}(|L|)$ Minkowski sum operations and $\mathcal{O}(|L|)$ intersection operations, so its running time is also $\mathcal{O}(|L|)$ in the complexity of these operations.

5.2. Stepping in reachable volume space

We define a method for stepping reachable volume samples to produce samples that are similar to the original (see Figure 19). This stepping function will serve as a primitive operation for a reachable volume local planner (Section 5.3) and a reachable volume RRT (McMahon et al., 2015).

This method starts with an initial configuration q_{RV}, a specified joint j, and a target configuration v_{RV}. It perturbs q_{RV} by moving j by δ in the direction of v_{RV} (Figures 19(b) and 19(c)). It then updates the placement of j and its descendants to ensure that all joints are in the reachable volume of their parents which will guarantee that the joint placements defined in the new sample q'_{RV} correspond to a constraint-satisfying configuration (Figures 19(d) and 19(e)).

We observe the following about repositioning a joint.

Observation 4. *If we reposition a joint j in such a way that it is still in the intersection of the reachable volumes of its parents, then only j's descendants need to be repositioned.*

Observation 5. *If we reposition a joint j and one of j's children is still in the intersection of the reachable volumes of both its parents, then all of the descendants of this child must also be in the intersection of the reachable volumes of their parents, and we do not need to reposition the child or its descendants.*

Observation 6. *If the original sample satisfied all joint placement constraints, then the final configuration must also satisfy all joint placement constraints.*

Proof. Every joint in the new sample is located in the reachable volume of that joint, which is a subset of any constraints on the placement of the joint. □

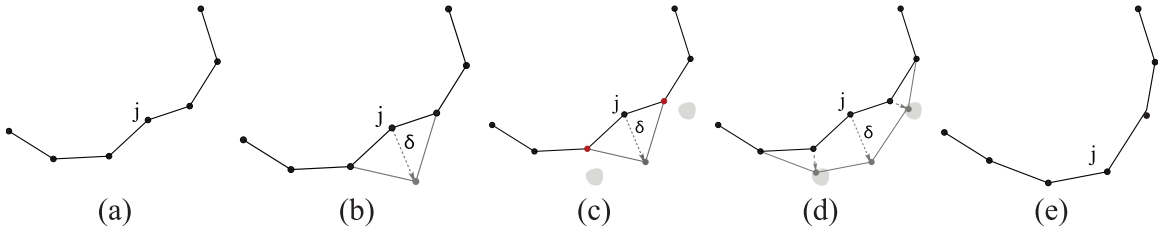

Fig. 19. Reachable volume stepping. We step one joint j by a distance δ (b) and then update the j's descendants to be in each of their reachable volumes given j's new placement (c). The gray regions are the reachable volumes of the third and fifth joints after j is stepped. These joints are repositioned to be in their reachable volumes (d) resulting in a configuration in which all joints are in their reachable volumes (e).

Algorithm 12 `RVStep`$(q_{RV}, j, p_{target}, \delta)$

Input: An RV-space configuration q_{RV}, a joint j, a target placement p_{target} of j, and a stepsize δ
Output: An RV-space configuration q'_{RV} in which the joint j has been perturbed by δ in the direction of p_{target}

Let p be the placement of j in q_{RV}
Let $C_1, C_2, \ldots, C_m$ denote the parent chains of j and $j_1, j_2, \ldots, j_m$ denote the neighboring joint to j in each parent chain
for $C_i \in C_1, C_2, \ldots, C_m$ **do**
    $RV_{C_i} =$ `ReachableVolume`(C_i)
$p' = p + \delta(p_{target} - p)$
if $p' \in RV_{C_1} \cap RV_{C_2} \cap \cdots \cap RV_{C_m}$ **then**
    $q'_{RV} = q_{RV}$ with j placed at p'
    for $C_i \in C_1, C_2, \ldots, C_m$ **do**
        `Reposition`(q'_{RV}, j_i) ▷ see Algorithm 13
    return q'_{RV}
return $\emptyset$

Algorithm 13 `Reposition`(q_{RV}, j)

Input: An RV-space configuration q_{RV} and a joint j
Output: An RV-space configuration such that j and all of its descendants are in the intersection of the reachable volumes of their parents

Let $C_1, C_2, \ldots, C_m$ denote the parent chains of j
Let $RV_{C_1}, RV_{C_2}, \ldots, RV_{C_m}$ denote the reachable volumes of each parent chain
if $j \in RV_{C_1} \cap RV_{C_2} \cap \cdots \cap RV_{C_m}$ **then**
    return q_{RV}
Let $q'_{RV} = q_{RV}$ with j adjusted to be within $RV_{C_1} \cap RV_{C_2} \cap \cdots \cap RV_{C_m}$
Let $j_1, j_2, \ldots, j_m$ denote the child joints of j
for $j_i \in j_1, j_2, \ldots, j_m$ **do**
    $q'_{RV} =$ `Reposition`(q'_{RV}, j_i)
return q'_{RV}

Algorithm 12 provides a method for stepping using reachable volumes based on these observations.

Based on Observation 1, we know that, with the exception of the children of j, all of the joints must still be located in the reachable volumes of their parents. We therefore only need to check whether the descendants of j are still within the reachable volume of their parents. To do this we recursively test the descendants of j (Algorithm 13). If a joint is no longer in the reachable volume of its parents, we reposition it and recurse on its children. If we encounter a joint that is still in the intersection of the reachable volume of its parents, then we can stop by Observation 2.

To reposition a joint, we move it to a placement that is in the intersection of the reachable volumes of the joint's parents and near the original placement of the joint. When repositioning a joint we know that all previously repositioned joints were placed in their reachable volumes, so there must exist a sample in this intersection. The reachable volume of a joint being repositioned will therefore never be empty and there will always be a valid repositioning. The result of repositioning is an RV-space configuration in which all joints are located in the intersection of the reachable volumes of their parents. Such a configuration must correspond to a feasible placement of the joints in the linkage.

By applying reachable volume stepping to an initial RV-space sample, we can create a new sample that is near the original. These samples can be generated randomly by selecting a random target point or they can be generated in a specific direction by selecting a target in that direction. There are also a number of ways to select what joint to perturb.

One of the advantages of reachable volumes is that they may be computed in any order. We observe that reachable volume stepping only effects the joint being perturbed and its children, meaning that the ordering will determine which joints are affected by the stepping operation. The following are some possible orderings which we will explore in our experiments.

- **Linear, end effector first:** construct the reachable volumes linearly with the end effector as the top. This limits the effects of stepping to the joints between the perturbed joint and the root. This ordering gives preference to end effectors, and would be useful for fixed base graspers and manipulators where the motion of end effectors is generally most significant.
- **Linear, root first:** construct the reachable volumes linearly with the root at the top. This limits the effects to

joints between the perturbed joint and the end effectors. This method steps a robot's internal joints starting at the root and would be useful for stepping closed chains or graspers with tight end-effector constraints.

- **Binary:** compute the reachable volumes in a binary manner (as described in McMahon et al. (2014b)). This localizes the effect of stepping to the children of the perturbed joint. This method produces the shortest, most direct overall paths and would be useful for navigating tight regions where longer paths have more chance of causing collisions.
- **Based on robot structure:** compute the reachable volumes so that related parts of the robot are in the same branch of the reachable volume tree. For example, a grasper robot could be partitioned so that the fingers are in separate branches of the tree. Consequently, perturbing a joint in one of the fingers will only effect joints in that finger. This method would be best for physical simulations (e.g., protein folding) where the importance of joints is determined by the physical structure of the robot.

5.3. Reachable volume local planner

We define a reachable volume local planner based on reachable volume stepping. As we will see in Section 7, this planner can be used by PRMs to find local paths that satisfy a problem's constraints.

The reachable volume local planner (Algorithm 14) connects two configurations, q_{RV1} and q_{RV2}, by using reachable volume stepping to move each joint to its placement in the second configuration. To accomplish this, it performs a traversal of the joints based on a given ordering (see Section 5.2). During each iteration of the traversal, it uses reachable volume stepping to move the joint from its placement in q_{RV1} to its placement in q_{RV2}.

Figure 20 is an example of the reachable volume local planner (with a binary reachable volume ordering) applied to a 5 link chain. The local planner first steps the end effector of the robot from its placement in q_{RV1} to its placement in q_{RV2} (Figure 20(a)). It then steps the third joint (Figure 20(b)), then the fourth joint (Figure 20(c)) to their placements in q_{RV2}. This results in the configuration q_{RV2} (Figure 20(d)). Note that we always step parents in the reachable volume traversal ordering before their children to ensure that stepping a joint will not change the placement of any joints that have already been stepped.

The sequence of steps covered by the local planner forms a path from q_{RV1} to q_{RV2}. We can test the validity of this path by checking the validity at each step in the same manner as with other local planners. If the robot is free-based, then we can use reachable volume sampling to generate paths between the internal joints of q_{RV1} and q_{RV2} and apply a rigid-body local planner to the translational and rotational coordinates. In most cases, we interleave the reachable volume local planner with the rigid-body local planner so that we perform part of the rigid-body transformation, apply the reachable volume sampler to the internal joints, and perform the rest of the rigid body transformation. This is analogous to the rotate-at-S local planner (Amato et al., 2000).

Algorithm 14 `RVLocalPlanner`$(q_{RV1}, q_{RV2}, \delta)$

Input: RV-space configurations q_{RV1} and q_{RV2} and a step size δ

Output: A sequence of configurations P from q_1 to q_2 where q_1 and q_2 are the C-space configurations corresponding to q_{RV1} and q_{RV2}, $P = \emptyset$ if no path is found

1: $P_{RV} = \emptyset$
2: Let j_{base} denote the first joint in the traversal ordering (see Section 5.2)
3: Let Q be a queue, push j_{base} onto Q
4: **while** $j =$ popped joint from Q **do**
5: $p_{target} = j$'s placement in q_{RV2}
6: $q'_{RV} = q_{RV1}$
7: **while** j's placement in $q'_{RV} \neq p_{target}$ **do**
8: $q'_{RV} =$ `RVStep`$(q'_{RV}, j, p_{target}, \delta)$
9: **if** q'_{RV} is $\emptyset$ or q'_{RV} is not valid **then**
10: **return** $\emptyset$
11: **else**
12: Add q'_{RV} to P_{RV}
13: **for** each child j_{child} of j **do**
14: Push j_{child} onto Q
15: Let P be the sequence of RV-space configurations in P_{RV} converted to C-space
16: **if** P is valid by a C-space rigid-body local planner **then**
17: **return** P
18: **else**
19: **return** $\emptyset$

Reachable volume stepping returns a sequence of configurations in RV-space. This sequence satisfies the constraints on the system but may not be collision-free. This sequence must be converted to a sequence of C-space configurations that can be checked for collision. This is done in line 16 of Algorithm 14. Note that if the resulting sequence contains consecutive configurations that are too far apart to assume that the motion between them is collision-free (i.e., they exceed the collision checking resolution), this pair of configurations must be interpolated at a higher resolution in RV-space and then converted back to C-space until the collision checking resolution is met. The C-space sequence of configurations is checked for collisions as in traditional sampling-based motion planning in C-space. This implementation detail is removed from the pseudocode for clarity.

5.4. Reachable volume distance metric

We define a reachable volume distance metric that measures the distance traversed during reachable volume stepping.

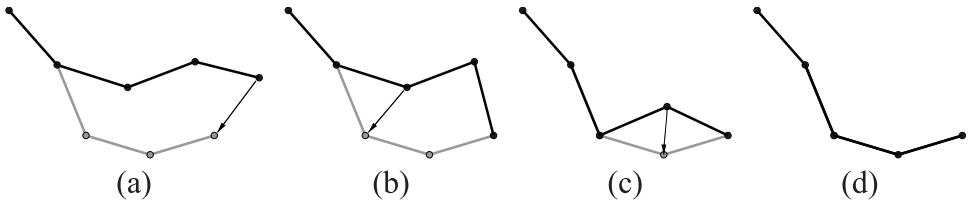

Fig. 20. The reachable volume local planner uses reachable volume stepping to move the joints of a linkage from their placements in one configuration (black) to their placements in a second (gray).

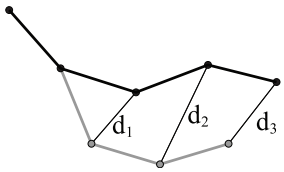

Fig. 21. The reachable volume distance between two samples (black and gray) is the sum of the distances between the joints of the configurations in reachable volume space. Here the reachable volume distance is $d_1 + d_2 + d_3$.

The reachable volume local planner constructs paths by moving each of the joints from its placement in the first configuration to its placement in the second configuration, so a good approximation would be the sum of the distances between the joints in reachable volume space (Figure 21). For free-base systems, a distance metric must also take into account the translational and rotational distance between configurations. This can be accomplished by adding the translational and rotational distance to the reachable volume distance (with a scaling factor, s):

$$D_{tran+rv}(q_1, q_2) = (s)\,\text{Euclidean}(base_{q_1}, base_{q_2}) + (1-s)\sum_{j \in J} \text{Euclidean}(j_{q_1}, j_{q_2})$$

where j_{q_1} and j_{q_2} are the placements of j in q_1 and q_2 in RV-space, $base_{q_1}$ and $base_{q_2}$ are the position and orientations of the base in q_1 and q_2, and s is a scaling factor.

6. Reachable volume PRM

In this section we describe a reachable volume PRM that uses reachable volume sampling in combination with the reachable volume local planner and reachable volume distance metric. We then evaluate the coverage and distribution of reachable volume samples, and show that reachable volume PRM is probabilistically complete.

6.1. Method overview

The reachable volume PRM uses the reachable volume sampler, reachable volume local planner, and reachable volume distance metric in the PRM framework presented in Kavraki et al. (1996). The reachable volume PRM (Algorithm 15) constructs a roadmap (N, E) by iteratively generating samples, q, using reachable volume sampling. If q is valid, the method adds it to the roadmap and then attempts to connect it to the k closest roadmap nodes using the reachable volume local planner. If a valid local plan is found, then it adds an edge between those nodes to the roadmap.

Algorithm 15 `RVPRM`$(R, env, done, k)$

Input: A robot R, and environment *env*, an evaluator *done*, and an integer k
Output: A roadmap (N, E)

```
1:  N = ∅
2:  E = ∅
3:  while done do
4:      c = SampleRVxxx(R) where SampleRVxxx refers to the appropriate sampling method for the robot type
5:      if q is valid in env then
6:          Add q to N
7:          Let N_q be the k closest neighbors to q by RVDistance ▷ as described in Section 5.4
8:          for all n ∈ N_q do
9:              P = RVLocalPlanner(q, n)
10:             if P ≠ ∅ then
11:                 Add (q, n) to E
12: return (N, E)
```

6.2. Coverage and sample distribution

Coverage describes a method's ability to generate samples over a problem's sample space. Methods that are capable of generating samples in a region of sample space are said to cover that region, whereas methods that are capable of generating samples over the entire sample space are said to cover the sample space. If a method does not cover the sample space, it will be unable to solve problems that require samples from regions it does not cover, no matter how many samples are generated. Coverage is therefore a necessary condition for probabilistic completeness of any motion planning sampler.

Sample distribution describes how densely or sparsely regions of sample space are sampled which is determined by the probability distribution of the sampling method. Methods with a skewed probability distribution will oversample some regions of sample space while producing few

samples in other regions. If a critical region (e.g., a narrow passage) lies in the region that is poorly covered, then a method will require a large number of iterations to generate enough samples in the region to solve it. Such a method would not be able to solve the problem efficiently. It is therefore important that a sampling method generates samples over a problem's entire samples space, and especially important that it generates samples in the critical regions. Unlike most methods that sample in C-space, reachable volume sampling samples in RV-space. We will therefore evaluate the coverage and sample distribution in RV-space as well as in C-space.

6.2.1. Coverage of reachable volume samples. We first observe that coverage of RV-space is equivalent to coverage of C-space for fixed-base robots. For free-base robots, coverage of RV-space is equivalent to coverage over the internal dofs. Consequently, if a method disjointly covers both RV-space and the external (translational/rotational) dofs, then it covers C-space. Reachable volume sampling samples a problem's external dofs using uniform sampling which covers the external degrees of freedom with a uniform sample distribution. Consequently RV-space coverage implies C-space coverage in both free- and fixed-base problems. We can therefore evaluate the coverage of reachable volume sampling in RV-space with the knowledge that our analysis also holds to coverage in C-space.

6.2.2. Distribution of reachable volume samples. C-space and RV-space sample distributions are not equivalent. C-space samples consist of a parameter for each dof and their distribution is the distribution of these parameters. For internal dofs, this is the distribution of the angles of the joints. Reachable volume space configurations consist of the placement of each of the joints of the robot in RV-space. The distribution of RV-space samples is therefore the distribution of these joint placements. We can approximate this distribution as the distribution of the distances between the pairs of joints in RV-space.

Unlike sampling in joint space where each dof is independent of each other, sampling one reachable volume affects the available range of the other unsampled reachable volumes, by design. The distribution of reachable volume samples is therefore dependent on the order in which the joints of the robot are sampled. The sample distribution of the first joint of the robot is uniform across the reachable volume of that joint and the reachable volume of each subsequent joint is uniform across its reachable volume given the placement of all of the previously sampled joints (see Figure 22).

In Section 5.2 we proposed an end-effector-first ordering and a root first ordering. We discuss the distribution resulting from each ordering in turn.

Recall that an *end-effector-first* ordering performs a depth-first traversal of the joints of the robot sampling each joint on the return of this traversal. As such it samples an end effector of the robot first and samples joints only after all of their descendants have been sampled. In unconstrained problems, the volume of reachable volumes is either quadratic (2D workspace) or cubic (3D workspace) with respect to distance from the root (see Figure 23). The number of samples within a distance d of the root is proportional to the volume of a circle/sphere of radius d over the volume of whole circle/sphere. The probability density of samples will therefore be skewed towards the larger portion of the feasible distance range. Consequently, the end-effector-first method will tend produce elongated samples in which the distances between joints and the root are skewed toward the upper portion of their feasible range and distances from joints to their end effector are skewed towards the lower portion of their distance range.

In addition, recall that a *root first* ordering performs a pre-order traversal of the robot starting at the root. This method samples each joint then recurses on each of its unsampled neighbors. As such, it always samples a joint prior to sampling its children and consequently samples joints that are closer to the root before those that are further from the root. For unconstrained robots the method will produce a similar sample distribution to uniform sampling.

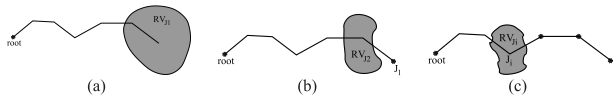

Fig. 22. The probability distribution of the first joint sampled (a) is uniform of the reachable volume of that joint (RV_{J1}). The probability distribution of the second joint sampled (b) is uniform over the reachable volume of that joint (RV_{J2}) given the placement of the first joint sampled. The probability distribution of the ith joint sampled, J_i is the reachable volume of J_i (RV_{Ji}) given the placement of all joints that were sampled prior to J_i. Note that black circles correspond to the root and to joints that have already been sampled.

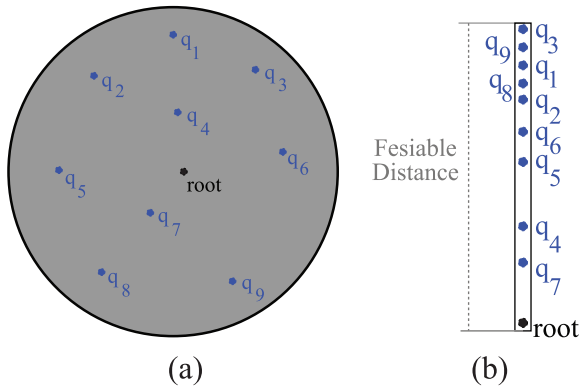

Fig. 23. Sampling joints uniformly over a (2D) reachable volume (a) will result in a sample set that is skewed towards the upper portion of the feasible distance range (b).

6.3. Probabilistic completeness

In this section, we discuss the sampling distribution of the reachable volume sampler and show that it is probabilistically complete.

Lemma 4. *Joints are sampled uniformly in their reachable volume given the placement of the joints that are already sampled.*

Proof. The joint sampling methods presented in Section 5.1.3 uniformly sample a domain that contains the reachable volume until they find a sample in the reachable volume resulting in a distribution that is uniform in the reachable volume. □

Lemma 5. *Reachable volume sampling is probabilistically complete.*

Proof. The samplers iterate through a robot's joints and sample them in their reachable volume (the region they can reach given the placement of the joints already sampled). The joint sampling methods sample over the entire reachable volume of a joint so we can inductively conclude that all possible reachable volume space configurations can be sampled. There is a one-to-one correspondence between reachable volume samples and joint angle settings. Consequently, the reachable volume sampler is complete over the range of joint angle coordinates. Our method uses the probabilistically complete reachable volume sampler to sample any joint angle coordinates and a uniform sampler (which is also probabilistically complete) to sample any translational and rotational coordinates resulting in a probabilistically complete sampler. □

7. Evaluation of reachable volumes

In this section we evaluate the reachable volume framework on a set of unconstrained and constrained problems. Our experimental results are organized in the following manner. In Section 7.1 we discuss our experimental setup and introduce the environments used. In Section 7.2 we evaluate reachable volume sampling, in Section 7.4 we evaluate the reachable volume local planner and reachable volume distance metric, and in Section 7.4 we evaluate the quality of roadmaps produced using reachable volume techniques.

7.1. Experimental setup

We first give a description of our experimental methodology and an overview of the environments used in our experiments.

7.1.1. Methodology. When designing our experiment set we took into consideration the experiment sets in papers that address similar problems, particularly Yershova et al. (2005); Yershova and LaValle (2009), Jaillet and Porta (2011), and Suh et al. (2011). The experiment set in Yershova et al. (2005); Yershova and LaValle (2009) includes a set of closed chains with as many as 20 dofs and an environment with a bolt on a chain. The experiment set in Jaillet and Porta (2011) includes a point on a torus environment, an environment with a five-link arm, and an environment with a planar manipulator with two arms holding an object (similar to our wheeled grasper). The experiments in Suh et al. (2011) consist of a rigid-body environment, a 7-dof arm robot, and a 12-dof closed-chain environment that is similar to our wheeled grasper.

We include linkages and closed chains that are similar to the linkage and closed-chain problems in these papers (the only difference being that we use spherical joints whereas they used articulated joints). We also include much higher-dof versions of the closed-chain and linkage environments (we include problems with as many as 262 dofs whereas they only studied problems with 20 dofs). Our experiment set also includes linkages and closed chains in environments with narrow passages (the tunnel and walls environments) whereas the previous studies ran experiments either in totally free environments or in cluttered environments where there is a large amount of free space between obstacles.

We implemented all planners using the C++ motion planning library developed by the Parasol Lab at Texas A&M University, which uses the graph from the STAPL Parallel C++ library (Tanase et al., 2011). All computations were performed on Brazos, a major computing cluster at Texas A&M University. The processing nodes consisted of quad-core Intel Xeon processors running at 2.5 GHz, with 15 GB of RAM. All experiments had a maximum time allocation of 20 hours. Results are averaged over 10 runs.

7.1.2. Problems studied. We ran experiments using chains, tree-like graspers, and closed-chain robots in the following environments. The combinations of environments and robots we used are listed in Table 2.

Walls: The walls environment (Figure 24(a)) consists of three chambers separated by two walls. In this environment, we run experiments with free flying chain linkages of varying dofs (22–262) and single-loop closed chains of (22–72 dofs).

Tunnel: The tunnel environment (Figure 24(b)) consists of two chambers connected by a long narrow tunnel. We run experiments using free flying chain linkages and single loop closed chains of varying dofs (22–262).

Grid: The Grid environment (Figure 24(c)) consists of a set of cube obstacles arranged in a grid. We run experiments using a 16-link (32-dof) fixed-based chain, a 16-link (32-dof) fixed-base grasper, and a 32-link (64-dof) fixed-base grasper.

WAM clutter: The WAM clutter (W-CL) environment (Figure 24(d)) contains a Barrett WAM robotic arm surrounded by a clutter of obstacles. The WAM arm consists of a 6-dof arm with three graspers attached to it (15 dofs in

Table 2. Combinations of environments and robots used in our experiments

Environment	Robot type	dofs
Walls	Chain	22, 70, 134, 262, 518, 1,034
Tunnel	Chain	22, 70, 134, 262
Grid	Chain	32
Grid	Tree-like Robot	32, 64
WAM clutter (W-CL)	Tree-like robot	15
Free	Closed Chain	22, 70, 262
Walls	Closed Chain	22, 70, 262
Tunnel	Closed Chain	22, 70, 262
Rods	Closed Chain	22, 70
Wheeled grasper (Wh-gr)	Closed Chain	19, 67
Loop-tree (Lp-tr)	Multi-loop Closed Chain	160
Cord	Constrained Chain	16, 64
Bug-trap cleaner (bt)	Constrained Chain	16
Fixed base grasper (gr)	Constrained Tree-like Robot	32, 64
WAM Bars (w-b)	Constrained Tree-like Robot	15, 22
Constrained closed chain (cc)	Constrained Closed Chain	22, 70
Wheeled grasper with bucket (wb)	Constrained Closed Chain	22

total). This robot is interesting in that it includes both planar and spherical joints, demonstrating that our method is applicable to robots that include different types of joints.

Free: We run closed-chain experiments of varying dofs (22–262) in a free environment.

Rods: The rods environment (Figure 24(e)) consists of four rods. Closed chains may enclose the rods and move onto different rods through breaks in them. In this environment we used 22- and 70-dof single-loop closed chains.

Loop-tree robot: The loop-tree (Lp-tr) robot (Figure 24(f)) consists of an eight-link central loop with four eight-link branches attached to it. At the end of each branch another eight-link loop is attached giving this robot a total of five loops and 160 dofs. Experiments are run in a completely free environment.

Wheeled grasper: We study a wheeled robot with two graspers attached to it (Figure 24(g)). The graspers have spherical joints and need to transport an object under a low hanging environment, thus forming a closed chain. We study a 19- and a 67-dof robot.

Robot with cord: The robot with cord environment (Figure 24(h)) consists of a chain robot with a cord attached to one of its joints. The robot's motion is constrained by the length of the cord so that the distance between the joint and the base of the cord cannot exceed the length of the cord (light gray region). We use two variations of this environment, one with 16 dofs and one with 64 dofs.

Fixed-base grasper: The fixed-base grasper environment (Figure 24(i)) consists of a fixed-base tree-like robot whose end effectors are constrained to be grasping one of the obstacles in the environment (green region). We include results for a 32-dof variation of this environment (gr-32) and a 64-dof variation (gr-64). This environment demonstrates that our method can be applied to grasping problems.

Constrained closed chain: The constrained closed-chain environment (Figure 24(j)) consists of a 22-dof closed chain (cc-22) or a 70-dof closed chain (cc-70). Constraints are applied to three of the chain's joints so that these joints must always be within a small distance of each other.

Wheeled grasper with bucket: The grasper with bucket (Figure 24(k)) is a variation of the wheeled grasper environment in which the grasper is carrying a bucket that must remain level with the ground. We include results for a 22-dof variation of this environment (wb-22).

Bug-trap cleaner: The bug-trap cleaner (bt) environment (Figure 24(l)) is a variation of the bug-trap benchmark in which a 16-dof fixed-base robotic arm must clean out the bug-trap. The base of the arm is located outside of the bug-trap while the end effector of the arm is constrained to be inside.

WAM bars: The WAM bars (w-b) environment (Figure 24(m)) consists of a Barrett WAM robotic arm that consists of a 6-dof arm with three graspers attached to it (15 dofs in total). The graspers are constrained to be grasping an object that is separated from the robot by a set of bars. The robot must reach through the bars to grasp the object.

7.2. Evaluation of reachable volume sampling

In this section, we evaluate the reachable volume sampler in a set of constrained and unconstrained problems. We study the time required to generate valid samples as well as the proportion of samples that are valid. Our results show that reachable volume samples are less likely to contain self-collisions and that reachable volume sampling requires less time to generate samples in high-dof problems.

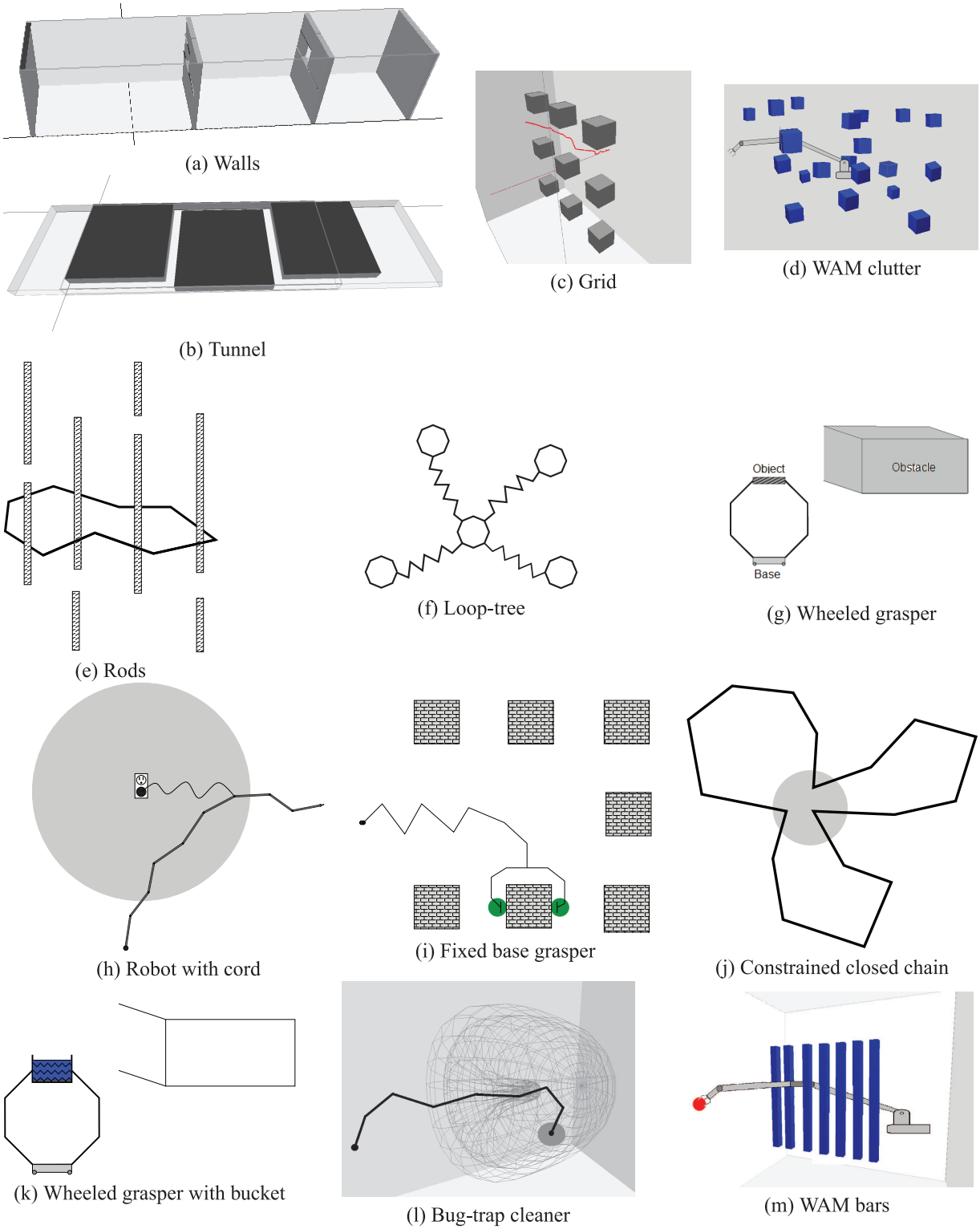

Fig. 24. Environments studied.

7.2.1. Comparison sampling methods. We compare reachable volume sampling (Section 5.1) to uniform sampling (Kavraki et al., 1996) and an incremental sampling method, I-CD, developed for these experiments.

I-CD is similar to uniform sampling except that it interleaves sampling and collision detection. This is particularly helpful for long chains by preempting samples that have early collisions. I-CD detects invalid links as soon as they

Algorithm 16 `IncrementalCDSampling`(R)

Input: A robot R containing a set of joints J

Output: A valid randomly sampled configuration q that specifies a value for each dof or $\emptyset$ if the attempt is invalid

```
1:  Randomly sample the translational and rotational dofs of the base, record base dof values in q
2:  if base placement collides with any obstacles in the environment then
3:      return ∅
4:  else
5:      for j ∈ J do
6:          Sample joint angles for j and record in q
7:          for each child link l of j do
8:              if l collides with any previously sampled links or obstacles in the environment then
9:                  return ∅
10:     return q
```

are sampled eliminating the need to sample the rest of the chain when collisions are found (see Algorithm 16).

Uniform sampling and I-CD sampling can be applied to chains and tree-like robots, however neither method can generate samples that satisfy closure constraints associated with closed-chains. For closed chains we also compare with a CCD sampler (Wang and Chen, 1991), which uses CCD to produce closed-chain configurations. In problems with constraints we compare to CCD as well as to a uniform sampler that filters out samples that do not satisfy a problem's constraints.

CCD can be applied to single-loop closed chains and problems where constraints are placed on a single end effector, however it cannot handle multi-loop robots, constraints on internal joints or constraints on multiple joints/end effectors. Unfortunately, there has been very little work in motion planning for linkages and closed chains with spherical joints. All of the other methods presented in our related work are only applicable to robots with single-dof joint angles. Aside from uniform sampling, CCD, and I-CD sampling, we are not aware of any existing sampling methods for linkages with spherical joints.

7.2.2. For chains and tree-like robots. In this section we evaluate the performance of reachable volume sampling for chains and tree-like robots. Our results show that reachable volumes requires less time to generate samples than other methods, particularly in high-dof problems.

Figure 25 compares the performance of reachable volume sampling, uniform sampling, and I-CD in generating 2,000 valid samples for various environments with chains and tree-like robots. Stars indicate methods that were unable to generate 2,000 samples in the allotted 20 hours (e.g., uniform sampling for the tunnel environment with more than 70 dofs or for the walls environment with more than 134 dofs).

The sampler success rate (Figure 25(a)) is the proportion of samples that are valid (e.g., collision-free). This indicates how efficient a method is at generating valid samples which can be used for roadmap construction. In lower-dimensional problems, uniform sampling and I-CD have higher success rates than reachable volume sampling. This is because the distribution of reachable volume samples results in more collisions with obstacles (i.e., external collisions). However, as the number of dofs of the problem increases, reachable volume sampling outperforms the other methods, and in some cases is the only method able to generate valid samples in the allotted time. Interestingly, the success rate of reachable volume sampling does not significantly decrease with problem dimension.

Figure 25(b) provides the time required for each method to generate 2,000 valid samples. We see that reachable volume sampling is slower than the others in lower-dimensional problems such as the 22-dof chains. This is a result of the overhead associated with computing reachable volumes and the lower sampler success rate in these problems. In higher-dimensional problems, the time is considerably better because reachable volume samples are more elongated (see Section 6.2.2) and, thus, less likely to have self-collisions which become more problematic as the robot complexity increases. In the highest-dimensional problems shown, only reachable volume sampling was able to complete within the allotted time (20 hours).

7.2.3. For closed chains. We next evaluate reachable volume sampling for closed chains. Our results demonstrate that reachable volume sampling requires less time to generate samples than other methods, especially in higher-dof problems. They also show that reachable volume sampling is able to generate samples in many environments where other methods fail.

Figure 26 gives the performance of reachable volume sampling for robots containing closed chains. Again, 2,000 valid, constraint-satisfying samples are created for each problem. Neither uniform sampling nor I-CD are able to generate constraint-satisfying samples for any of the robots in the time allotted. Only reachable volume sampling can handle systems with spherical joints.

As expected, the sampler success rate decreases as problem complexity increases (Figure 26(a)), yet reachable volume sampling is still able to generate valid, constraint-satisfying samples for single loops up to 262 dofs and complex robots such as the loop-tree with 160 dofs. This trend is echoed in the increasing time required to generate such samples (Figure 26(b).

In comparison to CCD sampling, reachable volume sampling consistently produced samples that were more likely to be successful and required considerably less time to generate successful samples. In the rods 134- and 262-dof free environments, the CCD sampler did not finish in the allotted time of 20 hours while the reachable volume sampler successfully generated samples. Moreover, reachable volume

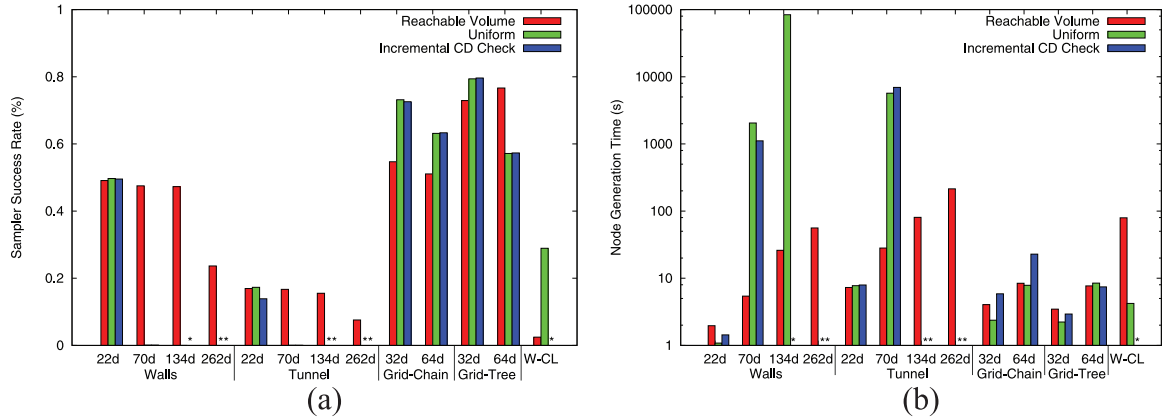

Fig. 25. Experimental results for chains and tree-like robots in various environments to generate 2,000 valid samples: (a) sampler success rate; (b) node generation time. Stars indicate methods that were unable to generate samples in the allotted time. Note that (b) uses a log scale. Environments are walls, tunnel, grid, and WAM clutter (W-CL) (see Figure 24(a)–(d)).

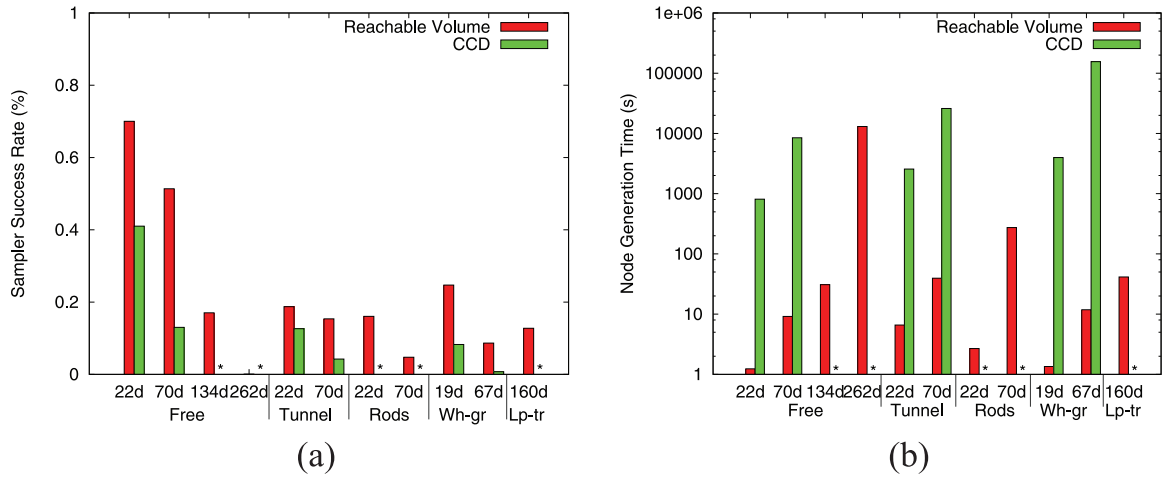

Fig. 26. Experimental results for closed chains in the following environments to generate 2000 valid samples: free, tunnel, rods, wheeled grasper (Wh-gr), and the loop-tree robot (Lp-tr). (a) Sampler success rate; (b) node generation time. Uniform sampling and I-CD are infeasible for these robots. Note that (b) uses a log scale. Environments are free, tunnel, rods, wheeled grasper (wh-gr), and loop-tree (lp-tr) (see Figure 24(b), (e)–(g)).

sampling can be applied to problems such as the loop-tree robot where CCD sampling is cannot.

7.2.4. For constrained systems. We next evaluate reachable volume sampling on a set of constrained systems. Again, we show that reachable volume sampling requires less time to generate samples than other methods and that reachable volume sampling is able to generate samples in environments where other methods fail.

Figure 27 gives the performance of reachable volume sampling for constrained systems. Again, 2,000 valid, constraint-satisfying samples are created for each problem. Reachable volume sampling is the only method able to generate samples for every problem in the allotted time, and in those problems where other methods do generate samples, reachable volume sampling almost always requires less time.

Our results show that reachable volume samples are more likely to be valid than samples produced by other methods (Figure 27(a)). They also show that reachable volume sampling is able to produce samples in difficult environments where other methods fail.

Our timing results (Figure 27(b)) show the running time of reachable volume sampling is less than uniform sampling that filters constraint-violating samples and significantly less than CCD. This is because uniform sampling with filtering generates many samples that must be discarded because they do not satisfy constraints, and CCD requires significant time to step samples towards constraints. In comparison, reachable volume sampling always generates samples that satisfy constraints without the need for any expensive stepping operations. The sole exception to this trend is the bug-trap environment where uniform sampling with filtering has a lower running time.

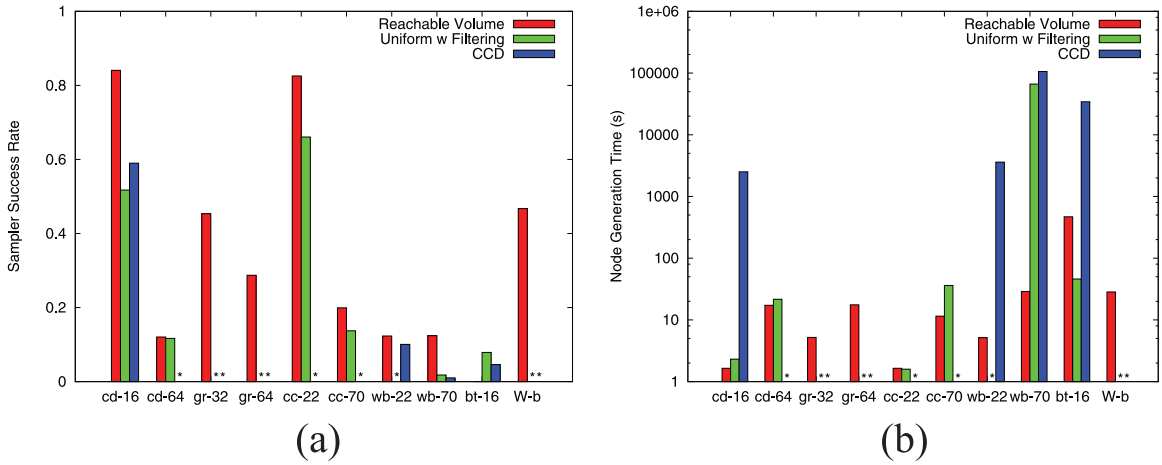

Fig. 27. Experimental results for 2,000 samples in various constrained systems: (a) sampler success rate; (b) node generation time. Stars indicate methods that were unable to generate samples in the allotted time or were not applicable. Note that (b) uses a log scale. Environments are robot with cord (cd), fixed base graper (gr), constrained closed chain (cc), wheeled grasper with bucket (wb), bug-trap cleaner (bt), and WAM bars (W-b) (see Figure 24(h)–(m)).

In the grid and WAM environments, both uniform sampling with filtering and CCD fail to generate samples. This is interesting because it demonstrates one of the shortcomings of these methods. The grid and WAM environments both consist of tree-like graspers with constraints applied to their end effectors. Although a method like CCD can converge to a single end-effector constraint, neither CCD nor uniform sampling with filtering can ensure that the base of the fingers of the grasper is placed where the other graspers can reach their associated constraints. In these problems you need a more powerful method such as reachable volume sampling that can place the base of the graspers such that all of the end effectors can reach their constraints. The CCD method also fails in the higher-dof cd-64 and cc-22/cc-70 environments.

7.2.5. Coverage and sample distribution. We next study the distribution of reachable volume samples to show that reachable volume sampling produces good coverage. Our results demonstrate that reachable volume sampling produces similar and in many cases better coverage than existing methods such as uniform sampling and CCD (Wang and Chen, 1991).

Reachable volume sampling samples positional and rotational coordinates in the same manner as uniform sampling, resulting in a similar sample distribution to uniform in these dimensions. We therefore focus on the distribution of the internal portion of samples. We evaluate internal configurations by evaluating the distances between pairs of joints within the samples, and we study sample distribution by studying the distribution of these distances.

We studied coverage and sample distribution in four different environments: a four-link open chain, a four-link open chain with links of unequal length, a five-link closed chain, and a four-link chain with an end-effector constraint (see Figure 28). We studied the coverage and distribution of two variations of reachable volume sampling, one with an end-effector-first ordering and one with a root-first ordering (see Section 5.2). In the open-chain environments, we compared with uniform sampling, whereas in the closed chain and constrained environment, we compared with CCD.

To evaluate coverage and sample distribution, we generate 100 samples using each method. For each sample we then compute and plot the Euclidean distance between each pair of non-adjacent joints (Figures 29, 30, 31, and 32). These figures plot the individual pair-wise distances between non-adjacent joints for each sampled configuration (i.e., there are 100 distances plotted for each joint pair, one distance for each sampled configuration). Studying the range of distances produced and comparing them with the feasible range of distances for each joint pair will indicate how well each method covers the robot's sample space. Ideally, a method should produce samples over the entire range of feasible distances for all pairs of non-adjacent joints. The distances plotted between each pair of joints should ideally be distributed uniformly over the entire range of feasible distances.

Figure 29 shows a scatter plot of the distances separating all pairs of non-adjacent joints in a four-link open chain consisting of 0.25 unit links connected by spherical joints (see Figure 20(a)). We first observe that reachable volume sampling with an end-effector-first ordering covers the same range of distances as uniform sampling. The distribution of these distances is more uniform and consistent than uniform sampling. This is especially noticeable in the distances between joints J_0 and J_4 (denoted as (J_0, J_4) in Figure 29), where reachable volume sampling with end effector first ordering gives better coverage over the larger distance ranges (0.6 to 1). Overall, these results are consistent with the expected probability distribution of an end-effector-first ordering, which we describe in Section 6.2.2. These results

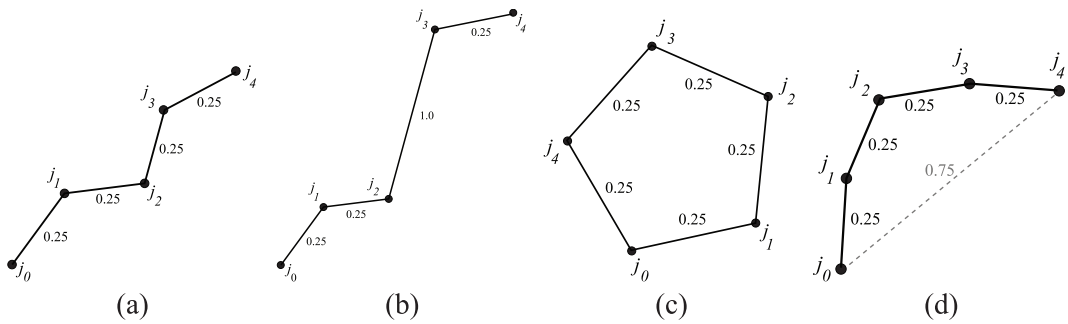

Fig. 28. Robots used in the joint distance study: (a) four-link chain; (b) chain with unequal link lengths; (c) five-link closed chain; (d) chain with end-effector constraint.

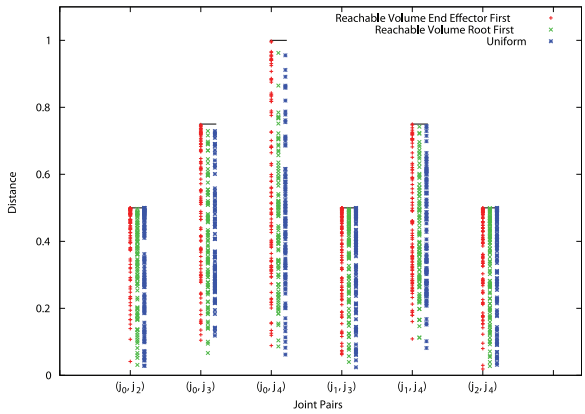

Fig. 29. Distance distribution between pairs of joints for 100 reachable volume/uniform samples of a four-link open chain with links of length 0.25 (Figure 28(a)). Horizontal lines indicate the maximum feasible distance between each joint pair.

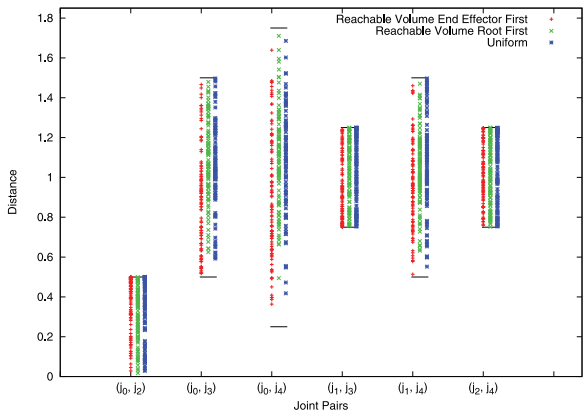

Fig. 30. Distance distribution between pairs of joints for 100 reachable volume/uniform samples of a 4 link open chain where the first, second and fourth links have a length of 0.25, and the third link has a length of 1 (Figure 28(b)). Horizontal lines indicate the maximum and minimum feasible distances between each joint pair.

indicate that an end-effector-first ordering produces better coverage over the regions of sample space that correspond to elongated or fully extended samples, which would be beneficial in problems where the robot needs to reach an object that is just inside its maximum range. Reachable volume sampling with root-first ordering also produces a similar distribution to uniform with the exception of joint pair (J_0, J_4). For joint pair (J_0, J_4) it obtains similar or better coverage over the middle of the distance range (0.2 to 0.8) but worse coverage outside of this range.

Figure 30 is a plot of the distances for a four-link open chain (Figure 28(b)) in which the first, second, and fourth links are 0.25 units long, and the third link is 1 unit long. Overall, both reachable volume sampling methods give better coverage than uniform sampling. Reachable volume sampling with end-effector-first ordering gives substantially better coverage in the lower distance ranges, as can be seen in the distances between (J_0, J_3), (J_0, J_4), and (J_1, J_4). Reachable volume sampling with root-first ordering gives better coverage of the upper distance ranges of (J_0, J_4) but worse coverage of the upper range of (J_1, J_4). Outside of this its coverage is similar to that of uniform sampling.

Figure 31 is a plot of distances for a single-loop closed chain composed of five 0.25 unit links connected by spherical joints (Figure 28(c)). We observe that both reachable volumes methods give better coverage of the lower distance ranges than CCD. This is particularly noticeable for end-effector-first ordering joint pairs (J_0, J_3) and (J_1, J_4) and the root-first ordering joint pairs (J_0, J_2) and (J_1, J_4). These results show that reachable volume sampling gives better coverage over the regions of sample space that correspond to highly deformed samples (where non-adjacent joints of the closed chain are close together).

We also observed that CCD produces samples that are outside of the feasible distance ranges. Recall that CCD generates samples by generating a random sample, then stepping it towards a problem's constraints until the sample is within a small value, ϵ, of the constraint. For closed chains such as the one we are studying, CCD first samples the chain as an open chain and then steps the end effector of the chain towards its root until the distance is less then ϵ. Consequently, CCD samples will not be fully closed and some of the samples will contain joints which are outside of the feasible range dictated by the closure constraint. Moreover, the running time of CCD increases drastically as

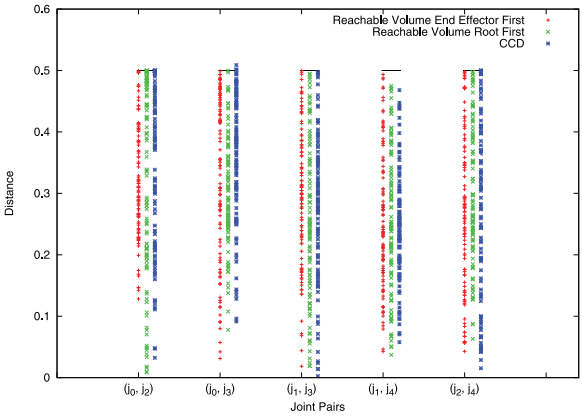

Fig. 31. Distance distribution between pairs of joints for 100 reachable volume/uniform samples of a five-link closed chain with links of length 0.25 units (Figure 28(c)). Horizontal lines indicate the maximum feasible distance between each joint pair.

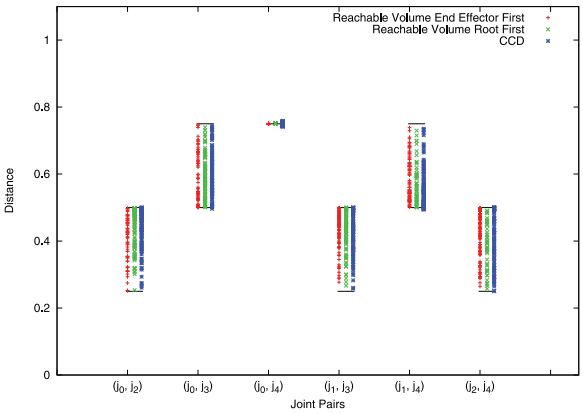

Fig. 32. Distance between pairs of joints for 100 reachable volume/uniform samples of a four-link open chain with links of length 0.25 with its end effector constrained to be 0.75 units from its base. Horizontal lines indicate the maximum and maximum distance between each joint pair.

epsilon approaches 0, making it unfeasible to generate samples that are more exact then those shown (e.g., with an ϵ value of 0.015). In contrast, reachable volume sampling can efficiently generate samples that exactly satisfy a problem's constraints.

Figure 32 is a plot of distances for a chain of four 0.25 unit links whose end effector is constrained to a point that is 0.75 from the root of the chain (Figure 28(d)). We observe that both reachable volume sampling methods produced distributions that were similar to that of CCD. The only exception was the root first joint pair (J_0, J_2) distance which did not cover the lower portion of the distance range as well as CCD. As in the closed chain experiments, CCD produced a small number of samples that were slightly outside of their feasible range.

7.3. Reachable volume local planner and distance metric in practice

In this section, we evaluate the reachable volume local planner and reachable volume distance metrics. We first study the performance of these methods in a set of sample environments. We then compare the connections produced by the reachable volume local planner with those produced by straight line, and the connections produced using the reachable volume distance metric with those produced using scaled Euclidean.

7.3.1. Performance. The primary advantage of the reachable volume local planner is that it generates paths that satisfy the problem's constraints whereas other methods such as straight line local planning do not. We compare the reachable volume local planner and reachable volume distance metric with straight line local planning and scaled Euclidean distance in the context of PRM construction. We use the walls environment as a case study and look at a 22-dof linkage (no constraints) and a 70-dof closed chain (has constraints). We attempt k-closest connection for 2,000 samples with $k = 8$ using the following combinations: reachable volume local planning with scaled Euclidean distance (rv-se), reachable volume local planning with reachable volume distance (rv-rv), and straight line local planning with scaled Euclidean distance (sl-se).

Figure 33 summarizes the results. For the unconstrained problem (w-22), rv-se and sl-se produce a similar number of edges using a similar amount of time. The reachable volume distance metric (rv-rv) does not perform as well here. For the constrained problem (w-cc), only rv-se and rv-rv are applicable as straight line local planning does not enforce the closure constraints. As in w-22, rv-se requires less time and produces more edges than rv-rv.

7.3.2. Connectivity. We next study the connectivity of roadmaps generated using the reachable volume local planner (rvlp) and distance metric (rvdm). We show that the reachable volume local planner can successfully connect many node pairs that straight line cannot. We also show that the candidate neighbors found by the reachable volume distance metric are significantly different than those found by scaled Euclidean, resulting in significant differences in roadmap connectivity.

We ran experiments in the walls environment (Figure 24(a)), which we ran in combination with a 22-dof chain (w-22) and a 22-dof closed chain (w-cc), and the tunnel environment (Figure 24(b)), which we ran in combination with a 22-dof chain (t-22). In each of these problems we generated 2,000 reachable volume samples. We then connected these samples to their eight nearest neighbors using different combinations of local planners and distance metrics, resulting in roadmaps with the same nodes but different edges. The combinations of methods and environments used are presented in Table 3. Note that we did not run

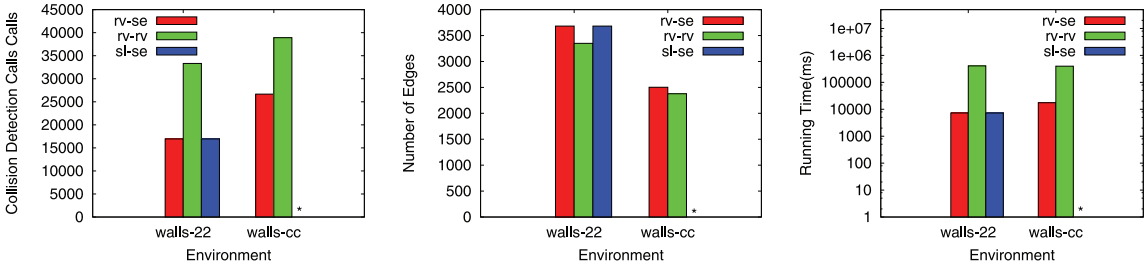

Fig. 33. (a) Collision detection calls, (b) number of edges, and (c) connection time for roadmaps constructed in the walls environment (see Figure 24(a)) using reachable volume local planning with scaled Euclidean (rv-se), reachable volume local planning with reachable volume distance (rv-rv), and straight line local planning with scaled Euclidean distance (sl-se). Stars denote when sl-se is not applicable.

Table 3. Combinations of local planner and distance metric used in each environment

	Straight line	rvlp
Scaled Euclidean	w-22, t-22	w-22, t-22, w-cc
rvdm	N/A	w-22, w-cc

experiments using the reachable volume distance metric in combination with straight line because the reachable volume distance metric was designed explicitly for use with the reachable volume local planner, and would likely not produce good results when combined with straight line. In addition, note that we did not run experiments using straight line for closed chains because straight line cannot generate paths that maintain closure constraints, making it unsuitable for closed chains.

Reachable volume local planner: We first compare the edges in roadmaps generated using the reachable volume local planner (rvlp) to those in equivalent roadmaps generated using straight line (sl). Figure 34 shows the number of edges present in both roadmaps along with the number of edges that are unique to each roadmap. These results show that both methods make a significant number of connections which the other method misses. These results indicate that it might be beneficial to use reachable volume local planing in combination with straight line when straight line is applicable.

Reachable volume distance metric: We next compare roadmaps generated using the reachable volume distance metric (rvdm) to those generated using scaled Euclidean (se). Our results (Figure 35) show there is a significant difference in the edges, which indicates that neighbors being selected by the reachable volume distance metric and significantly different from those selected by scaled Euclidean. Based on these results it might be beneficial to use a hybrid distance metric which combines the reachable volume distance metric and scaled Euclidean.

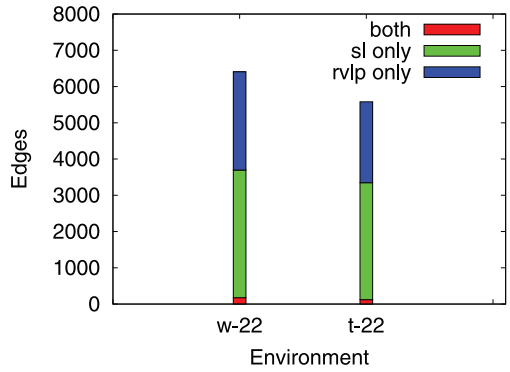

Fig. 34. Edge difference between reachable volume local planning (rvlp) and straight line (sl) using scaled Euclidean distance in the walls and tunnels environments (see Figure 24(a) and (b)).

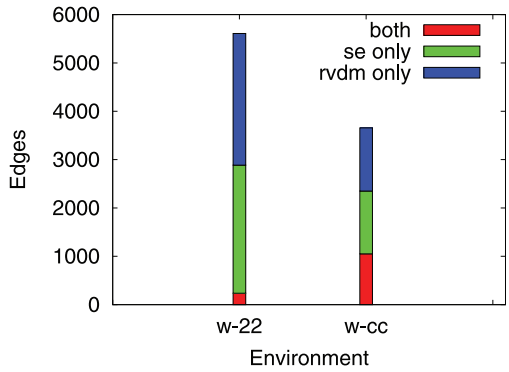

Fig. 35. Edge difference between reachable volume distance (rvdm) and scaled Euclidean (se) using the reachable volume local planner in the walls environment (see Figure 24(a)).

7.4. *Evaluation of reachable volume methods for PRM construction*

In this section, we study how reachable volume methods can be applied to roadmap construction. We show that reachable volume methods produce better connected roadmaps than existing methods and that they are capable of solving many difficult high-dof problems that existing methods cannot.

To evaluate our method we construct roadmaps in various environments. Roadmaps are constructed using k-closest

neighbor selection for identifying node pairs to connect with $k = 8$ under scaled Euclidean distance unless otherwise stated. Collision detection is performed using RAPID (Gottschalk et al., 1996) and local planning is done using a binary straight line local planner (Kavraki et al., 1996).

We evaluate the roadmap quality produced by each method along with their associated cost. To evaluate roadmap quality, we study the sampler success rate, the local planner success rate, the size of the largest connected component, and the number of connected components in the roadmap.

Figure 36(a) shows the percentage of local planner calls that are successful. This directly determines the number of edges that can be added to the roadmap, which, in turn, impacts how well connected it is. The local planner success rate for reachable volume sampling is consistently higher than the other methods, which indicates that reachable volume sampling produces samples that are easier to connect. This is because the joint orientations of reachable volume samples are more uniform meaning that connecting them is less likely to result in self-collision. The performance difference in local planner success rate becomes more significant with increasing problem dimensionality. This trend is especially noticeable for tree-like robots where the local planner can fail because of collisions between the branches of the robot.

The size of a roadmap's largest connected component (i.e., the percentage of nodes it contains) indicates how well connected it is. It also directly affects the number of different queries the roadmap can solve. Thus, roadmaps with larger percentages of samples in the largest connected component are more desirable. Figure 36(b) shows that roadmaps using reachable volume sampling produce a greater percentage of samples in the largest connected component than the other methods. This suggests that reachable volume sampling is doing a better job of finding connections between various areas of the free C-space, such as between the different chambers in the walls environment. This trend is particularly noticeable with the 70-dof chains and the tree-like robots.

The number of connected components (Figure 36(c)) tells us how good the methods are at producing connected roadmaps. Our results show that reachable volume sampling produces roadmaps with far fewer connected components than the uniform sampling or I-CD sampling. As with the local planner success rate, this indicates that reachable volume samples are easier to connect than samples produced by the other methods.

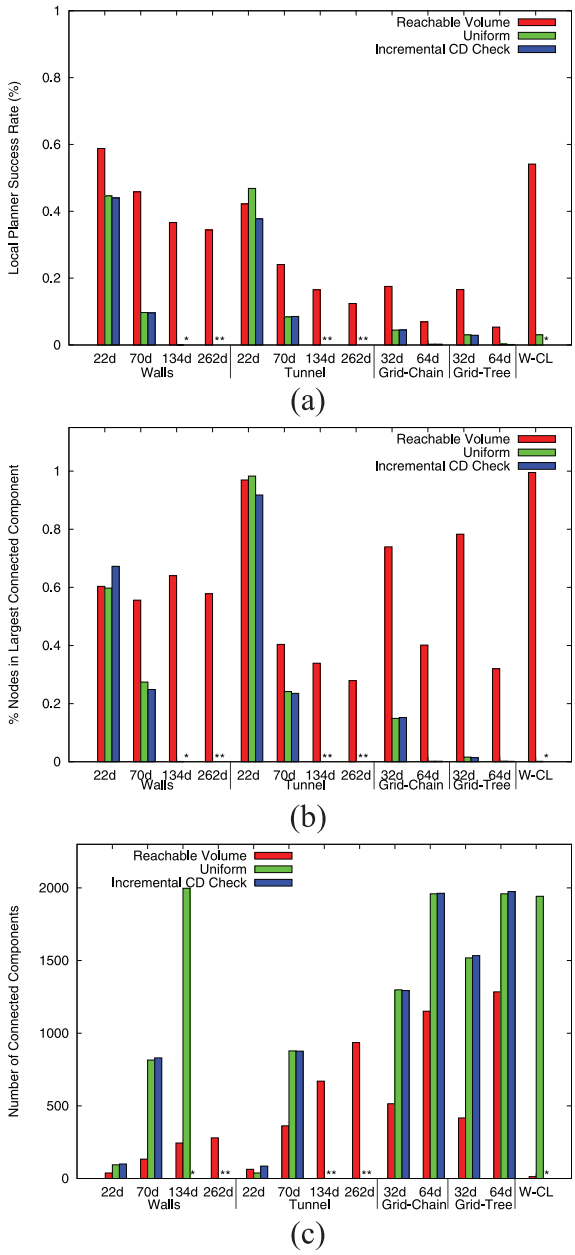

Fig. 36. Experimental results for chains and tree-like robots in various environments for 2000 samples: (a) local planner success rate; (b) percentage of nodes in the largest connected component; (c) number of connected components. Stars indicate methods unable to generate samples in the allotted time. Environments are walls, tunnel, grid, and WAM clutter (W-CL) (see Figure 24(a)–(d)).

7.4.1. Scalability of reachable volume PRMs. We next studied how the performance of reachable volume sampling scales with roadmap size. We studied the local planner success rate and largest connected component of roadmaps generated using the reachable volume sampler across a variety of roadmap sizes. These results show that reachable volume sampling can produce better connected roadmaps with fewer nodes than existing methods and that reachable volume sampling can generate samples and produce connected roadmaps in higher-dof problems than existing methods. They also demonstrate that reachable volume sampling can make connections through difficult narrow passages with fewer nodes and less execution time.

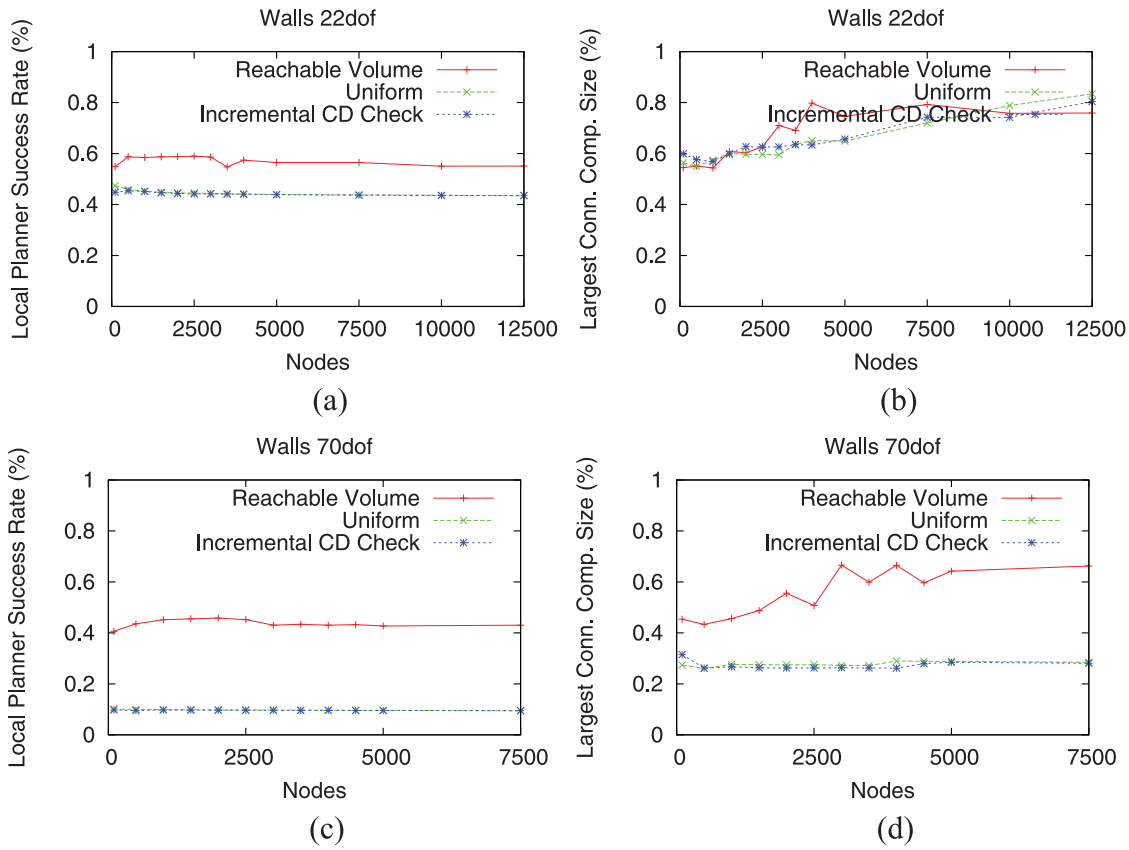

Fig. 37. Local planner success rate and size of largest connected component for 22- and 70-dof chains in the walls environment (see Figure 24(a)).

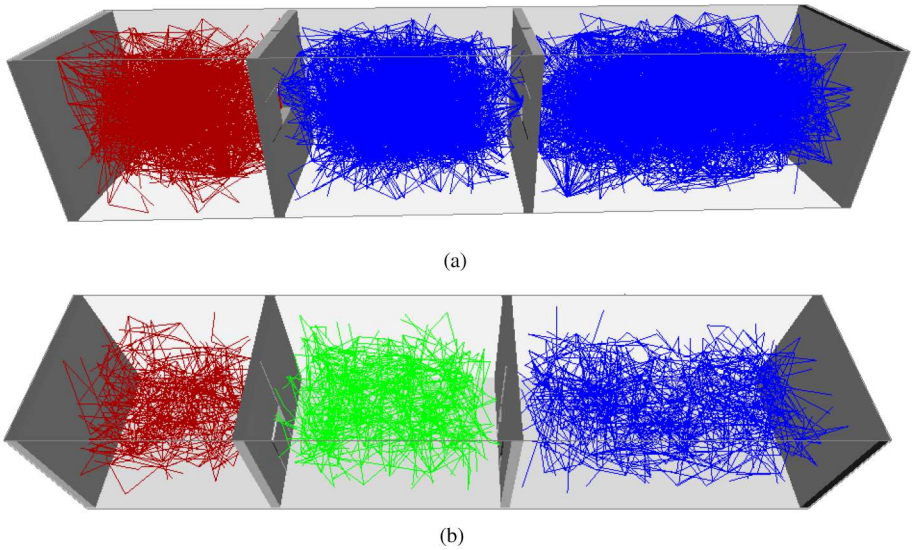

Fig. 38. Sample images of the connected components for the 70-dof chain in the walls environment with 3,500 samples. Note that the reachable volume sampler (top) connects two of the chambers whereas the uniform sampler (bottom) does not.

In the 22-dof walls environment (Figure 37(a) and (b)), the local planner success rate of both methods is significant, however the local planner success rate with reachable volume samples is consistently higher than with uniform sampling. The largest connected component size for both sampling methods is similar across the range of roadmap sizes we studied, and generally increases with larger roadmaps, indicating that adding more nodes is improving

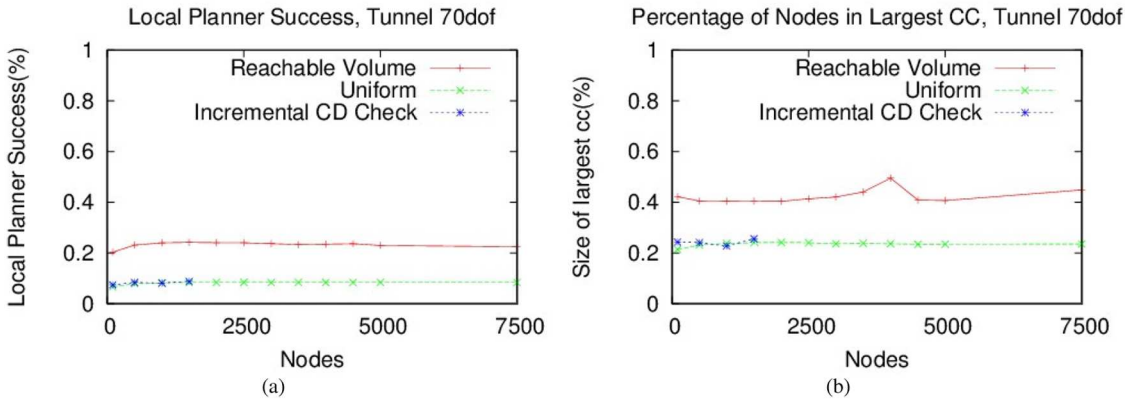

Fig. 39. Local planner success rate and size of largest connected component for the 70-dof chain in the tunnel environment (see Figure 24(b)).

Fig. 40. Sample image of the connected components for the 70-dof chain in the tunnel environment with 3,500 reachable volume samples. Note that this roadmap includes connected components in the tunnel region of the environment.

the connectivity for both methods. In the walls environment, the largest chamber contains 40% of the environment's free space and the other chambers contain 30%. Consequently, a largest connected component size that is larger than 40% implies that a method has connected two of the chambers and a largest connected component size that is greater that 70% implies that a method has connected all three chambers. Both methods produce roadmaps where the largest connected component size is significantly larger than 40%, which indicates that both methods are producing connections between chambers in the environment.

In the 70-dof walls environment (Figure 37(c,d)), the reachable volume sampler consistently produced roadmaps with a considerably higher local planner success rate than uniform sampling, even for large roadmap sizes. It also consistently produced roadmaps that are more connected than uniform sampling, resulting in larger largest connected components. The largest connected component size of the reachable volume roadmaps increased significantly over the range of roadmap sizes that we studied. For the larger values in this range, the largest connected component size is significantly larger than 40%, which indicates that it is consistently making connections between the chambers of the environment (as shown in Figure 38). In comparison, the largest connected component of the uniform sampling roadmaps remained at 30% over the entire range of roadmap sizes we studied.

In the 70-dof tunnel environment (Figure 39), the reachable volumes sampler also results in a higher local planner success rate than uniform sampling across the entire range of roadmap sizes we studied. The largest connected component size of the reachable volume roadmaps is significantly larger than uniform sampling over the entire range of roadmap sizes we studied. In the tunnel environment, 1/4 of the free volume is located in each of the free regions and 1/2 of the free volume is located in the tunnel region. The largest connected component size of the reachable volume roadmaps is significantly larger than 25%, which indicates that it is making connections into the tunnel (as demonstrated in Figure 40). The largest connected component size of the uniform roadmaps is never larger than 25%, which indicates that these components are confined to the free regions of the environments.

In higher-dof environments such as the walls environment with 262-dof, 518-dof, and 1,034-dof chains, and the tunnel environment with the 262-dof chain, Figure 41 demonstrates that the reachable volume sampler can be applied to high-dof motion planning problems. In these environments the reachable volume sampler successfully generates samples and produces roadmaps with a local planner success rate that is comparable with the lower-dof problems. The largest connected component size shows that these roadmaps are well connected and in the case of the walls environment, that they include connections between the chambers in the environment. Uniform sampling almost always generated samples with self-collisions and was not able to generate even 100 free samples before running out of time.

8. Conclusion

We have introduced a new concept, reachable volumes, that are a geometric representation of the regions the joints and end effectors of a robot can reach, and have used it to define a new planning space called RV-space where all points automatically satisfy a problem's joint placement constraints.

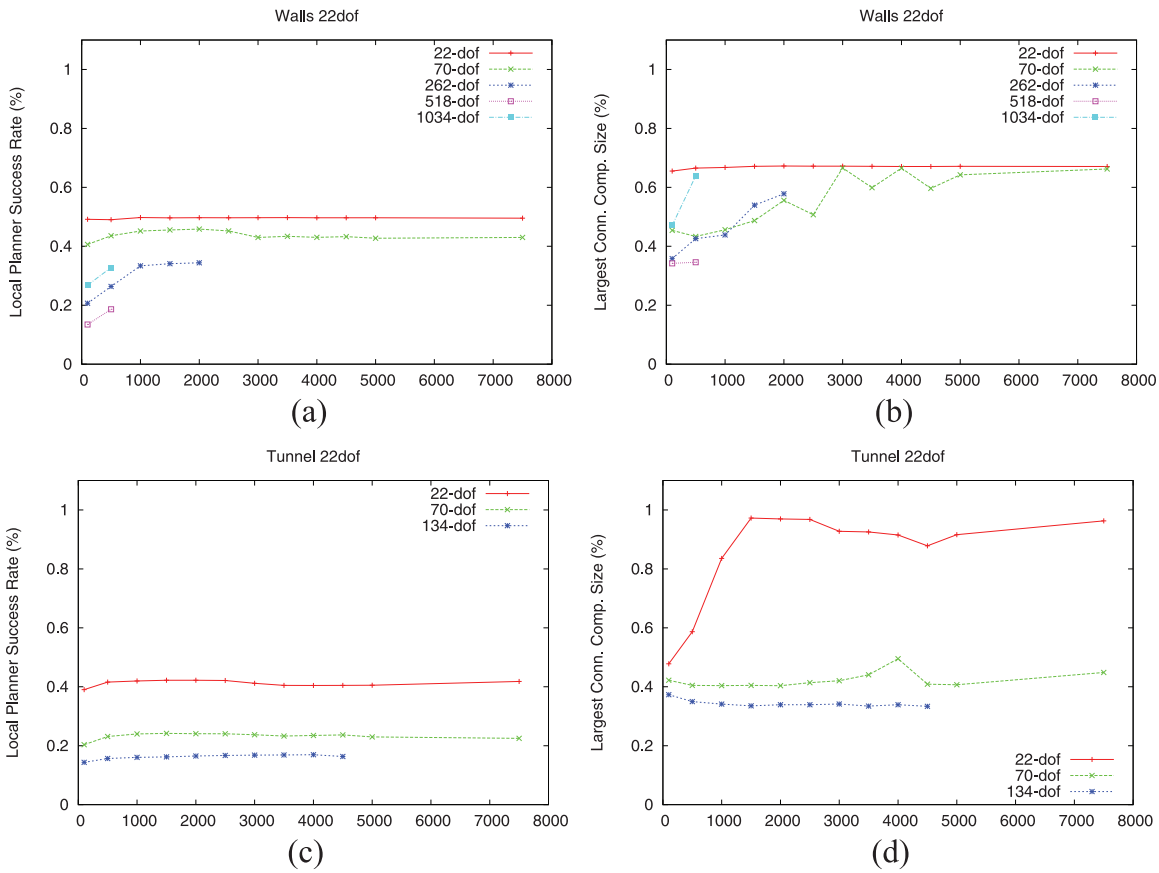

Fig. 41. Evaluation of how local planner success rate and size of the largest connected component scales with roadmap size in (a), (b) walls (see Figure 24(a)) and (c), (d) tunnel (see Figure 24(b)) environments with robots ranging from 22 to 1,034 dofs.

(Note that this does not include collision constraints.) Visualizations of reachable volumes can enable operators to see the regions of workspace that different parts of the robot can reach. Samples and paths generated in RV-space naturally conform to constraints, making planning for constrained systems no more difficult than planning for unconstrained systems. Consequently, constrained motion planning problems that were previously difficult or unsolvable become manageable and in many cases trivial. We have shown that reachable volumes have a $O(1)$ complexity and can be computed in $O(1)$ time in problems without constraints.

We have introduced tools and techniques to extend the state-of-the-art sampling-based motion planning algorithms to RV-space. We have presented a reachable volume sampler, a reachable volume local planner, and a reachable volume distance metric. These tools are applicable to robots with combinations of planar, prismatic, and spherical joints and to problems with constraints on the joints and end effectors of the robot. We have shown that the running time of the reachable volume sampler is linear with respect to the number of joints in the robot in unconstrained problems, and that it is linear in the complexity of the reachable volumes in problems with constraints. We have also shown that PRMs constructed using reachable volumes are probabilistically complete.

We have demonstrated that reachable volume sampling can be applied to a wide variety of problems including high-dof chains, tree-like linkages, closed chains, and combinations of them. We have presented results for constrained problems with as many as 70 dofs and for unconstrained problems with as many as 1,034 dofs. We have shown that the reachable volume sampler produces more connectable samples with greater ease than existing methods for constrained systems with spherical joints and with combinations of planar and spherical joints.

Funding

This research was supported in part by the NSF (award numbers CNS-0551685, CCF-0833199, CCF-1423111, IIS-0916053, IIS-0917266, EFRI-1240483, and RI-1217991) and the NIH (grant number NCI R25 CA090301-11).

References

Amato NM, Bayazit OB, Dale LK, Jones CV and Vallejo D (2000) Choosing good distance metrics and local planners for probabilistic roadmap methods. *IEEE Transactions on Robotics and Automation* 16(4): 442–447.

Anderson-Sprecher P and Simmons R (2012) Voxel-based motion bounding and workspace estimation for robotic manipulators.

In: *Proceedings IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2141–2146.

Cortés J and Siméon T (2003) Probabilistic motion planning for parallel mechanisms. In: *Proceedings IEEE International Conference on Robotics and Automation (ICRA)*, Taipei, Taiwan, pp. 4354–4359.

Cortés J and Siméon T (2005) Sampling-based motion planning under kinematic loop-closure constraints. In: *Algorithmic Foundations of Robotics VI*. Berlin: Springer, pp. 75–90.

Cortés J, Siméon T and Laumond JP (2002) A random loop generator for planning the motions of closed kinematic chains using PRM methods. In: *Proceedings IEEE International Conference on Robotics and Automation (ICRA)*, Washington, DC, pp. 2141–2146.

Craig JJ (1989) *Introduction to Robotics: Mechanics and Control* (2nd edn). Reading, MA: Addison-Wesley Publishing Company.

Garber M and Lin MC (2003) Constraint-based motion planning using Voronoi diagrams. In: *Algorithmic Foundations of Robotics V*. Berlin: Springer, pp. 541–558.

Gottschalk S, Lin MC and Manocha D (1996) OBB-tree: A hierarchical structure for rapid interference detection. *Computer Graphics* 30: 171–180.

Han L (2004) Hybrid probabilistic roadmap — Monte Carlo motion planning for closed chain systems with spherical joints. In: *Proceedings IEEE International Conference on Robotics and Automation (ICRA)*, New Orleans, LA, pp. 920–926.

Han L and Amato NM (2001) A kinematics-based probabilistic roadmap method for closed chain systems. In: *New Directions in Algorithmic and Computational Robotics*. Boston, MA: A. K. Peters, pp. 233–246.

Han L and Rudolph L (2007a) Inverse kinematics for a serial chain with joints under distance constraints. In: *Robotics Science and Systems II*. Cambridge, MA: MIT Press, pp. 177–184.

Han L and Rudolph L (2007b) A unified geometric approach for inverse kinematics of a spatial chain with spherical joints. In: *Proceedings IEEE International Conference on Robotics and Automation (ICRA)*, Roma, Italy, pp. 4420–4427.

Han L and Rudolph L (2009) Simplex-tree based kinematics of foldable objects as multi-body systems involving loops. In: *Robotics Science and Systems IV*. Cambridge, MA: MIT Press.

Han L, Rudolph L, Blumenthal J and Valodzin I (2006) Stratified deformation space and path planning for a planar closed chain with revolute joints. In: *Proceedings of the International Workshop on Algorithmic Foundations of Robotics (WAFR)*, New York, NY, pp. 235–250.

Han L, Rudolph L, Blumenthal J and Valodzin I (2008) Convexly stratified deformation spaces and efficient path planning for planar closed chains with revolute joints. *The International Journal of Robotics Research* 27(11–12): 1189–1212.

Han L, Rudolph L, Chou M, et al. (2012) Configurations and path planning of convex planar polygonal loops. In: *Proceedings of the International Workshop on Algorithmic Foundations of Robotics (WAFR)*.

Han L, Rudolph L, Dorsey-Gordon S, et al. (2009) Bending and kissing: Computing self-contact configurations of planar loops with revolute joints. In: *Proceedings IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1346–1351.

Jaillet L and Porta J (2011) Path planning with loop closure constraints using an atlas-based RRT. In: *15th International Symposium on Robotics Research*.

Kallmann M, Aubel A, Abaci T and Thalmann D (2003) Planning collision-free reaching motion for interactive object manipulation and grasping. *Computer Graphics Forum* 22(3): 313–322.

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.

Latombe JC (1991) *Robot Motion Planning*. Boston, MA: Kluwer Academic Publishers.

LaValle S, Yakey J and Kavraki L (1999) A probabilistic roadmap approach for systems with closed kinematic chains. In: *Proceedings IEEE International Conference on Robotics and Automation (ICRA)*, Detroit, MI, pp. 1671–1676.

LaValle SM and Kuffner JJ (1999) Randomized kinodynamic planning. In: *Proceedings IEEE International Conference on Robotics and Automation (ICRA)*, pp. 473–479.

LaValle SM and Kuffner JJ (2001) Rapidly-exploring random trees: Progress and prospects. In: *New Directions in Algorithmic and Computational Robotics*. A. K. Peters, pp. 293–308.

Lien JM (2008) Hybrid motion planning using minkowski sums. In: *Proceedings of the Robotics: Science and Systems Confrence (RSS)*.

Lozano-Pérez T and Wesley MA (1979) An algorithm for planning collision-free paths among polyhedral obstacles. *Communications of the ACM* 22(10): 560–570.

McMahon T, Thomas S and Amato NM (2014a) Sampling based motion planning with reachable volumes: Application to manipulators and closed chain systems. In: *Proceedings IEEE International Conference on Intelligent Robots and Systems (IROS)*. Chicago, Il., pp. 3705–3712.

McMahon T, Thomas S and Amato NM (2014b) Sampling based motion planning with reachable volumes: Theoretical foundations. In: *Proceedings IEEE International Conference on Robotics and Automation (ICRA)*, Hong Kong, China, pp. 6514–6521.

McMahon T, Thomas SL and Amato NM (2015) Reachable volume RRT. In: *Proceedings IEEE International Conference on Robotics and Automation (ICRA)*, Seattle, WA, pp. 2977–2984.

Merlet JP (2002) Still a long way to go on the road for parallel mechanisms. In: *ASME Biennial Mechanisms and Robotics Conference*, Montreal, Canada.

Milgram RJ and Trinkle JC (2004) The geometry of configuration spaces for closed chains in two and three dimensions. *Homology, Homotopy and Applications* 6(1): 237–267.

Oriolo G and Mongillo C (2005) Motion planning for mobile manipulators along given end-effector paths. In: *Proceedings IEEE International Conference on Robotics and Automation (ICRA)*, Barcelona, Spain, pp. 2154–2160.

Oriolo G, Ottavi M and Vendittelli M (2002) Probabilistic motion planning for redundant robots along given end-effector paths. In: *Proceedings IEEE International Conference on Intelligent Robots and Systems (IROS)*, Lausanne, Switzerland, pp. 1657–1662.

Reif JH (1979) Complexity of the mover's problem and generalizations. In: *Proceedings IEEE Symposium on Foundations*

of Computer Science (FOCS), San Juan, Puerto Rico, pp. 421–427.

Schneider R (1993) *Convex Bodies: The Brunn–Minkowski Theory*. Cambridge, MA: Cambridge University Press.

Suh C, Kim B and Park FC (2011) The tangent bundle RRT algorithms for constrained motion planning. In: *13th World Congress in Mechanism and Machine Science*.

Tanase G, Buss AA, Fidel A, et al. (2011) The STAPL parallel container framework. In: *Proceedings of the 16th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming (PPOPP 2011)*, San Antonio, TX, 12–16 February 2011, pp. 235–246.

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

Trinkle JC and Milgram RJ (2002) Complete path planning for closed kinematic chains with spherical joints. *The International Journal of Robotics Research* 21(9): 773–789.

Vargas Estrada A, Lien JM and Amato NM (2006) Vizmo++: a visualization, authoring, and educational tool for motion planning. In: *Proceedings IEEE International Conference on Robotics and Automation (ICRA)*, pp. 727–732.

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.

Whitney H (1932) Non-separable and planar graphs. *Transactions of the American Mathematical Society* 34: 339–362.

Whitney H (1933) 2-isomorphic graphs. *American Journal of Mathematics* 55: 245–254.

Xie D and Amato NM (2004) A kinematics-based probabilistic roadmap method for high DOF closed chain systems. In: *Proceedings IEEE International Conference on Robotics and Automation (ICRA)*, New Orleans, LA, pp. 473–478.

Yakey JH, LaValle SM and Kavraki LE (2001) Randomized path planning for linkages with closed kinematic chains. *IEEE Transactions on Robotics and Automation* 17(6): 951–958.

Yao Z and Gupta K (2005) Path planning with general end-effector constraints: using task space to guide configuration space search. In: *Proceedings IEEE International Conference on Intelligent Robots and Systems (IROS)*. Edmonton, Alberta, Canada, pp. 1875–1880.

Yershova A, Jaillet L, Simeon T and LaValle SM (2005) Dynamic-domain RRTs: Efficient exploration by controlling the sampling domaine. In: *Proceedings IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3867–3872.

Yershova A and LaValle SM (2009) Motion planning for highly constrained spaces. In: *Robot Motion and Control*. Berlin: Springer, pp. 297–306.

Zhang Y, Hauser K and Luo J (2013) Unbiased, scalable sampling of closed kinematic chains. In: *Proceedings IEEE International Conference on Robotics and Automation (ICRA)*,

Zucker M, Ratli N, Dragan AD, et al. (2013) CHOMP: Covariant hamiltonian optimization for motion planning. *The International Journal of Robotics Research* 32(9–10): 1164–1193.