SAMPLING BASED MOTION PLANNING WITH REACHABLE VOLUMES

A Dissertation

by

TROY ANTHONY MCMAHON

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Nancy M. Amato
Committee Members,	Ergun Akleman
	John Keyser
	Dezhen Song
Head of Department,	Dilma Da Silva

August 2016

Major Subject: Computer Science

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.

In this work, 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 provide tools and techniques to extend the state of the art sampling based motion planning algorithms to RV-space. We define a reachable volume sampler, a reachable volume local planner and a reachable volume 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+ degrees of freedom for multi-loop, and tree-like linkages.

DEDICATION

To my family who have helped and supported me through my academic career.

ACKNOWLEDGEMENTS

This dissertation and the work presented in it would not have been possible without the help and support of my family, friends and coworkers.

To my advisor Dr. Nancy Amato, I would like to thank you for your advise and assistance throughout my graduate career. You stood by me even through times of great frustration and I thank you for your patience and support. Your contribution to this work and to my academic success cannot be understated.

To my committee, Dr. Ergun Akleman, Dr. John Keyser and Dr. Dezhen Song, thank-you for the time you put into working with me. Your input, guidance and feedback were an important factor in my success as a student.

To Dr. Dylan Shell I thank you for filling in during my prelim. As with my committee members, your input and guidance were an important part of my success.

To Dr. Shawna Thomas, thank-you for all of the help and guidance, you were an important contributor to this work.

To my friends in the International Christian Fellowship, you were a continual source of peace, comfort and happiness during my time in Texas. Friday nights with you gave me something to look forward to each week. I will never forget you.

To my family including my dad Paul McMahon and his wife Daria. I am glad that I have had many opportunities to spend time with you over the past ten years. Our recent trips to Arizona and Gettysburg were both very memorable. I look forward to spending more time with you now that I am done with my dissertation.

To my mother Lorry McMahon and my sister Kara, even though I have not seen you in the last couple of years you are still in my thoughts.

To my cat Dusty, I would like to thank you for your constant affection and friendship. I wish that you had lived to see this day. I am grateful that I had the opportunity to see you one last time before you passed away.

To my extend family including my late grandparents and all of my aunts and uncles, I thank you to the contributions you have made to my life.

TABLE OF CONTENTS

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION

Constrained motion planning places constraints on the motion of an object (robot) and has applications in parallel robotics [32], grasping and manipulation [34], computational biology and molecular simulations [5], and animation [20]. 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 [7] 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.

Constraint satisfaction is a challenging problems and planning under constraints is particularly difficult because constraints must be satisfied over entire paths. Planning under constraints is especially difficult for problems where the constraints form a manifold in C-space, and planning must be restricted to this manifold (e.g., closed chains).

Sampling-based motion planning methods such as the graph-based PRM [21] and the tree-based RRT [23] are state of the art solutions to traditional motion planning problems. Unfortunately, because these methods rely on random sampling they cannot be applied to problems where constraints form manifolds because the probability of generating samples on a manifold is zero [25]. PRMs also rely on a local planner to connect samples and existing local planners are not able to generate paths on manifolds. Similarly, traditional RRTs cannot be applied because there is now way to expand the tree while ensuring that new nodes occur on constraint manifolds. Previous methods have developed specialized samplers that generate samples satisfying constraints [10, 15, 40] that can be used in combination with existing PRM-based and RRT-based methods to solve problems with constraints. However these methods are either unable to handle high degree of freedom (dof) systems or are unsuited for systems with spherical or prismatic joints or systems that combine different types of joints.

In this research, 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 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. Reachable volumes generalize the concept of reachable distances [40] 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 if 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 he can reach from a specified position, which can help him to decide where he 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 volume sampler, a reachable volume local planner, a reachable volume expansion function, and a reachable volume distance metric. Reachable volume sampling generates samples by iteratively sampling the joints of a robot in their reachable volumes, resulting in samples that are guaranteed to satisfy a problem's constraints. RV-based planners can solve problems with constraints applied to any combination of joints/end effectors, while most other methods (e.g. [43, 40, 10]) assume a single constraint, usually on one of the end effectors. The reachable volume local planner and distance metric can be 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 volume local planner, we present a novel method for stepping reachable volume 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 reachable volumes 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 diameter of the robot. 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 volume sampling produces more valid samples than existing methods, that reachable volume samples are easier to connect than other samples, and that reachable volume sampling is more efficient at solving high dimensional problems than existing methods. They confirm that its running time is linear with respect to the number of bodies in the robot. We show that the reachable volume local planner can produce constraint satisfying local paths with little overhead compared to the commonly used straight line local planner, which cannot find constraint satisfying paths.

The main contributions of this work include:

- 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 volume sampler, a reachable volume local planner, and a reachable volume 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 1034 dof.

This dissertation includes work from the paper "Sampling-Based Motion Planning with Reachable Volumes: Theoretical Foundations" published in the 2014 *IEEE Int. Conf. Robot. Autom. (ICRA)* [30] which introduced the concept of reachable volumes and presented tools and methods for applying them to sampling based motion planning. It also incorporates work from the paper "Sampling Based Motion

Planning with Reachable Volumes: Application to Manipulators and Closed Chain Systems" published in the 2014 *IEEE/RSJ Int. Conf. Intel. Rob. Syst. (IROS)* [29] in which we showed reachable volumes could be applied to motion planning for problems such as manipulators and systems of closed chains. Finally, it includes work from the paper "Reachable Volume RRT"published in the 2015 *IEEE Int. Conf. Robot. Autom. (ICRA)* [31] in which we presented a reachable volume local planner, distance metric and RRT. This dissertation 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 reachable volume configurations into C-space configurations, and an evaluation of how the reachable volume sampler scales with roadmap size.

The dissertation is organized in the following manner. In Chapter 2 we give an overview of related work and in Chapter 3 we define the set of problems that our work addresses. In Chapter 4 we introduce the concept of *reachable volumes* and show how to compute them. In Chapter 5 we present primitive operations including a reachable volume sampler, a reachable volume local planner, and a reachable volume distance metric. In Chapter 6 we present a reachable volume planner that uses these operations. In Chapter 8 we evaluate sampling based motion planning with reachable volumes.

2. RELATED WORK [*]

In this chapter 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 [36, 22]. Unfortunately, their running time is exponential with respect to the number of dof which makes them unsuitable for problems with more than 4 or 5 dimensions.

Sampling-based motion planning includes graph-based methods (e.g. Probabilistic Roadmaps (PRMs) [21]) and tree-based methods (e.g. Rapidly-Exploring Random Trees (RRTs) [24]). While PRMs and RRTs have been applied to a wide variety of problems, they both have been shown to be poorly suited for problems with spatial constraints [25]. 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 2.1 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

[*] Reprinted with permission from “Sampling based motion planning with reachable volumes: Theoretical foundations” by Troy McMahon, Shawna Thomas, and Nancy M. Amato, 2014. Proc. IEEE Int. Conf. Robot. Autom. (ICRA), pages 6514-6521, Copyright 2014 by IEEE.

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

PRMs and RRTs have been adapted for use in spatially constrained systems. *Gradient decent methods* push randomly generated configurations onto a constraint surface [25, 45]. These methods are capable of solving problems with single-loop, articulated joint, closed chains. PRM-MC combines a PRM and Monte Carlo methods to generate samples that satisfy closure constraints [13]. This method can efficiently generate samples for large (100 link) single-loop closed chains. In [41, 33], Trinkle and Milgram develop a method that uses C-space analysis for path planning while ignoring self collisions. They show 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 [46]. They present results for a 9 dof manipulator robot with a set of end effector constraints. In [49] Zhang et al. present 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 2D chains, closed chains and protein molecules with over 200 degrees of freedom.

There have been a number RRT based methods proposed for solving problems with constraints. DDRRT [47, 48] 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

[19] simultaneously builds an RRT and constructs an atlas, the set of charts which locally parametrizes constraint manifolds. The atlas is used to generate samples along the constraint manifolds, which are added to the RRT, while the RRT is used to guide the direction which the atlas is expanded. Tangent Bundle RRTs [38] construct an RRT along a set of tangent bundles which 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.

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 [10]. 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 [4, 5]. This method is shown to be faster than previous kinematics-based sampling methods. Kinematics-based methods have been extended to large linkages [44] and multiple loops [3].

Inverse kinematic methods for 3D, 5D, and 6D end effector constraints are shown to efficiently generate samples for chains with as many as 1000 links [15, 14]. They also present the concept of deformation space (D-space) which 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 [11, 12]. This method has been extended to produce paths guaranteed to be self-collision free [17]. 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.

While 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 decent (CCD) [43] moves the end effector of a robot to a specified end effector position by iteratively cycling through the robot's coordinates and adjusting them so that the end effector converges to the goal position. CCD can also be used to generate closed chain samples or samples which satisfy a specified end effector constraint for chains with as many as 7 dof.

CHOMP[50] 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 *h*ard constraints and optimize adherence to *s*oft constraints.

2.1.3 Enforcing Constraints During Sampling

Another approach is to explicitly enforce constraints while sampling. Han *et. al.* solve closed chain problems by transforming them into a system of linear inequalities [11]. Extensions are capable of solving closed chain problems with multiple loops [16]. 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 dof [8]. Other planners require the end effector to traverse a predefined trajectory by generating samples that satisfy the end effector constraints given by the trajectory [35, 34]. Han *et. al.* develop a method for generating samples with self-contact [18], i.e., configurations occurring on the border of C-free and the regions of C-obstacle that denote self-collisions. While 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 [2]. 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 [6] 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 [40]. 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. We extend 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.

Method	Constraint Types	High dof Robots	Closed Chains	Tree-like Robots	Joint Types	Multi-Query Problems
Basic PRM [21]	None	Yes	No	Yes	Planar, Spherical, Prismatic	Yes
Inverse Kinematics [10]	End Effector	Yes	Yes	Yes	Planar	Yes
Constrained Dynamics [8]	End Effector	Yes	Yes	No	Planar	Yes
I-CD	None	Yes	No	Yes	Planar, Spherical, Prismatic	Yes
DDRRT [47, 48],	Only End Effector Shown	Not shown	Yes	Not shown	Only Planar	No
Atlas RRT [19],	Only End Effector Shown	Not shown	Yes	Not shown	Only Planar	No
Tangent Bundle RRT [38]	Only End Effector Shown	Not shown	Yes	Not shown	Only Planar	No
CCD [43]	End Effector	No	Yes	No	Planar, Spherical	Yes
CHOMP [50]	Hard and Soft	Yes	Not Shown	Yes	Planar, Spherical, Prismatic, Combinations	No
Reachable Distances [40]	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**

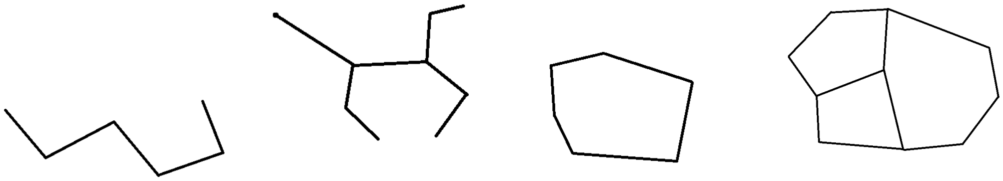
Table 2.1: Comparison of method capabilities.

3. PROBLEM FORMULATION: CONSTRAINED MOTION PLANNING WITH LINKAGES*

In this chapter we describe the types of linkage systems studied in this work and define 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 3.1(a)), or a tree, in which some of the joints will connect more than just two links (Figure 3.1(b)). Closed chain robots are a generalization of linkages in which chains of links may form one or many loops (Figure 3.1(c,d)).

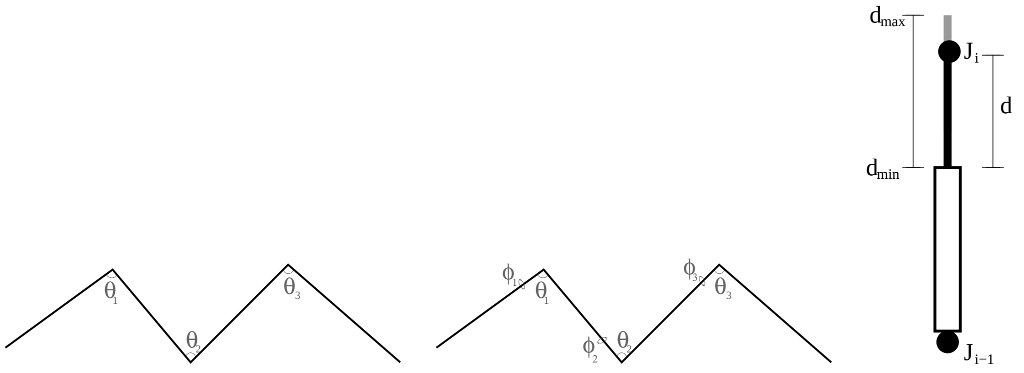


(a) Chain (b) Tree-like robot (c) Closed chain (d) Multiple loop closed chain

Figure 3.1: Examples of linkage systems.

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 joint

*Reprinted with permission from "Sampling based motion planning with reachable volumes: Theoretical foundations" by Troy McMahon, Shawna Thomas, and Nancy M. Amato, 2014. Proc. IEEE Int. Conf. Robot. Autom. (ICRA), pages 6514-6521, Copyright 2014 by IEEE.



(a) Planar joint

(b) Spherical joint

(c) Prismatic joint

Figure 3.2: (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}).

(see Figure 3.2(a)). Linkages connected by adjacent planar joints are coplanar and for chains comprised 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 3.2(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 3.2(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 *c*onfiguration space (of *C-space*) [28]. While it is not feasible in general to explicitly compute which portions of C-space are valid and which portions are not [36], 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. Closed chains also require that the chain remain closed throughout the motion of a path in order for it to be valid.

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 maintains 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

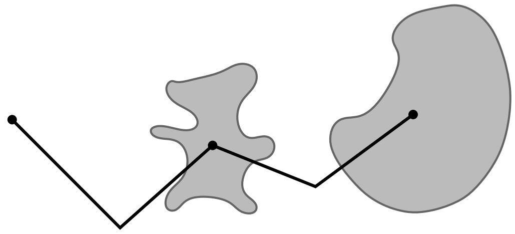


Figure 3.3: A chain linkage with constraints. Here, constraints are applied to the end effector and one of the internal joints of the chain as shown by the gray shaded regions.

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 3.3). 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 which 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.

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 [7] 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 chapter 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 a chain is the region of space that its end effector can reach. Below, 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 Volume Space

The *reachable volume space* (*RV-space*) of a linkage is a 3 dimensional 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. RV-space does not include any obstacles and can be used to generate sample configurations that are later tested for validity using a validity checker (e.g. [9]).

[*] Reprinted with permission from "Sampling based motion planning with reachable volumes: Theoretical foundations" by Troy McMahon, Shawna Thomas, and Nancy M. Amato, 2014. Proc. IEEE Int. Conf. Robot. Autom. (ICRA), pages 6514-6521, Copyright 2014 by IEEE.

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 4.1(a)).

First note that if the end effector can be positioned 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 the robot about the origin (see Figure 4.1(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 a distance of r from the base, where $r_1 \leq r \leq r_2$.*

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.

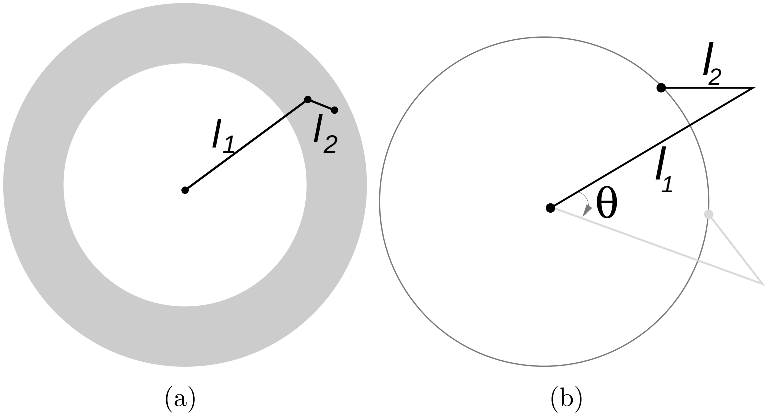


Figure 4.1: (a) The reachable volume (gray region) of a 2 link chain robot, l_1 and l_2 (black). l_1 rotates about the point in the center while l_2 rotates about the endpoint of l_1. (b) If the end effector (black) can reach a point, then it can reach all other points that are the same distance from the base (gray circle).

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}.

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

A RV-space configuration consists of a position in RV-space for each of the joints and end effectors. A configuration in RV-space is composed of a position for each joint and end effector in the robot. In a RV-space configuration, the position of a joint or end effector in RV-space is equal to the difference between the position of that joint or end effector in workspace and the position of the root in workspace. A RV-space configuration captures the relative position of the joints and end effectors.

A configuration in RV-space can be transformed into a C-space configuration by computing the joint dof 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 4.2(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 4.2(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 4.2(c)). Computing each θ and φ value can be done in constant time which means that a joint position configuration can be converted to a joint angle configuration in linear time with respect to the dof of the linkage.

- *Planar joints:* The position of adjacent joints must be coplanar. Thus, planar joints are a subset of spherical joints where φ is always 0. 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 [26] 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

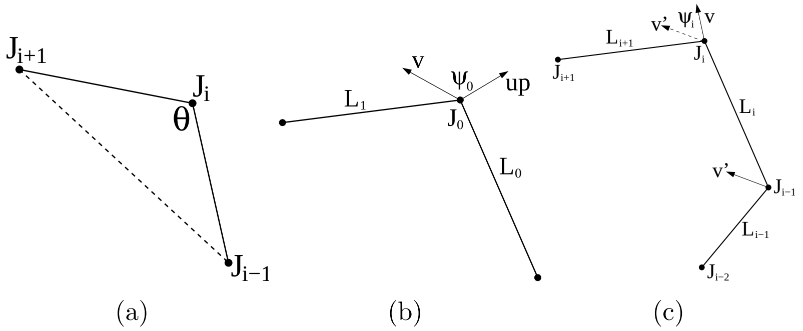


Figure 4.2: Computing joint angles: (a) We first compute the joint's articulated angle θ using the law of cosines. (b) The rotational angle φ_0 of the first joint is calculated by computing a vector v that is perpendicular to l_0 and l_1 and then computing the angle between this vector and the upward 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}. φ_i is the angle between v and v'.

coordinates are fixed. For each C-slice, they compute the Minkowski sum of the robot and the obstacles in the environment. They then sample along the boundary of the Minkowski sum and connect samples. Finally, they sort the C-slices and connect the nodes in nearby C-slices 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, then the reachable volume of C is equal to the Minkowski Sum of of the reachable volumes of C_1 and C_2.*

Proof. First, observe that a point in the 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 set, then the corresponding chain can reach a point that is (x,y,z) from the base of the chain (see Figure 4.3). 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 set of C_1 and the point (x_2,y_2,z_2) is in the reachable set of C_2, then the point (x_1+x_2,y_1+y_2,z_1+z_2) must be in the reachable set 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 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 in order for a point (x, y, z) to be in C, there must exist a point (x_1, y_1, z_1) in the reachable set of C_1 and a point (x_2, y_2, z_2) in the reachable set C_2 such that $x_1 + x_2 = x$, $y_1 + y_2 = y$ and $z_1 + z_2 = z$.

The reachable set of C is therefore the following:

$$\texttt{Reachable}(C) = \{(x_1 + x_2, y_1 + y_2, z_1 + z_2) \mid (x_1, y_1, z_1)$$

$$\in \texttt{Reachable}(C_1)\texttt{and}(x_2, y_2, z_2) \in \texttt{Reachable}(C_2)\}$$

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

$$\texttt{Reachable}(C) = \texttt{Reachable}(C_1) \bigoplus \texttt{Reachable}(C_2)$$

□

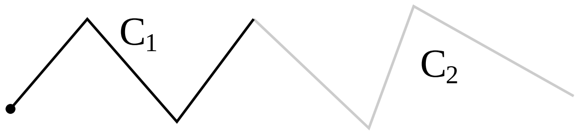


Figure 4.3: The reachable volume of a chain that is composed of two smaller chains C_1 (black) and C_2 (gray) is the Minkowski sum of the reachable volumes of C_1 and C_2.

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

$$\texttt{Reachable}(C) = \texttt{Reachable}(l_1) \bigoplus \texttt{Reachable}(l_2) \bigoplus \cdots \bigoplus \texttt{Reachable}(l_N)$$

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 chain is a sphere that is centered at the origin and has a radius equal to the length of the chain. The reachable volume of a chain is therefore the Minkowski sum of a set of spheres which can easily be computed (see Section 4.2.3). Note that Minkowski sums are commutative [37], which implies that the order in which the links occur in a chain has no impact on the reachable volume of the chain.

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

Algorithm 1 Compute Reachable Volume

```
Input: A chain $C$
Output: The reachable volume of $C$
if $C$ only has 1 link then
    return reachable volume of link
Let $j$ be an arbitrary internal joint from $C$
Let $C_l$ be the portion of $C$ to the left of $j$
Let $C_r$ be the portion of $C$ to the right of $j$
$RV_l$ =ReachableVolume($C_l$)
$RV_r$ =ReachableVolume($C_r$)
return $RV_l \bigoplus RV_r$
```

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 Chapter 4.4 and displayed using the Vizmo visualization tool [42].

Figure 4.4 shows the reachable volumes for each link in a simple 4 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 4.5, 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 the reachable volume space changes again when a chain is constrained to form a single loop. Figure 4.6 shows the reachable volume of our original 4-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) while the second joint can reach any point inside of the outermost sphere (blue).

Figure 4.7 shows the effect of constraining the end effector of the chain. Here the end effector of the chain is constrained to be 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).

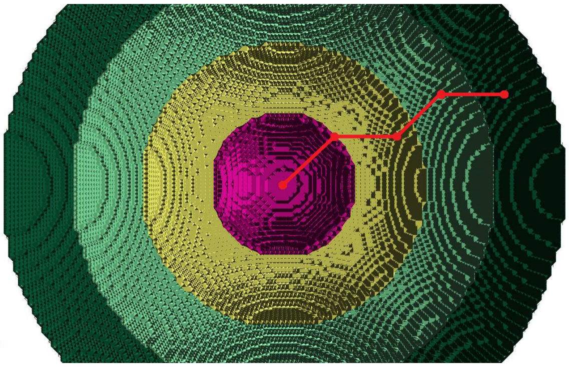


Figure 4.4: A cross section of the reachable volumes of a chain linkage (red) with 3 spherical joints, 4 links of equal length and no constraints. The first joint can reach any point along the inner sphere (pink), the second joint can reach any point inside the second sphere (yellow), the third joint 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.

Figures 4.8(a) and 4.8(a) show reachable volumes of a 16 dof fixed-base grasper with spherical joints in an environment with a set of cubic objects. Figure 4.8(a) shows the reachable volume of the base when the end effectors are constrained to spherical regions on either side of the object, while Figure 4.8(b) shows the reachable volumes of the end effectors when the base is constrained. Note that when the base is constrained in such a way, the end effectors each have the same reachable volume so only one is shown.

Figure 4.9 displays the reachable volume of a WAM robot [27] with 15 dof and

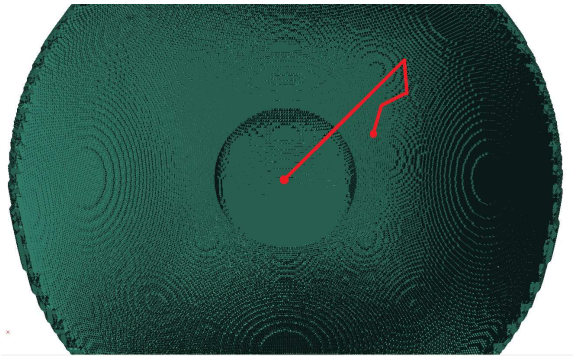


Figure 4.5: A cross section of the reachable volume of the end effector of a chain with 1 long link and 3 smaller links. The length of the three smaller links is less than the length of the first long link. The end effector can reach any point between the inner and outer spheres, but it cannot reach the region inside the inner sphere. An example configuration is shown in red.

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, and the wrist must be within the left region. The reachable volumes of the knuckles 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 chains, computing the reachable volumes of the chains, and then merging them to form the reachable volume 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

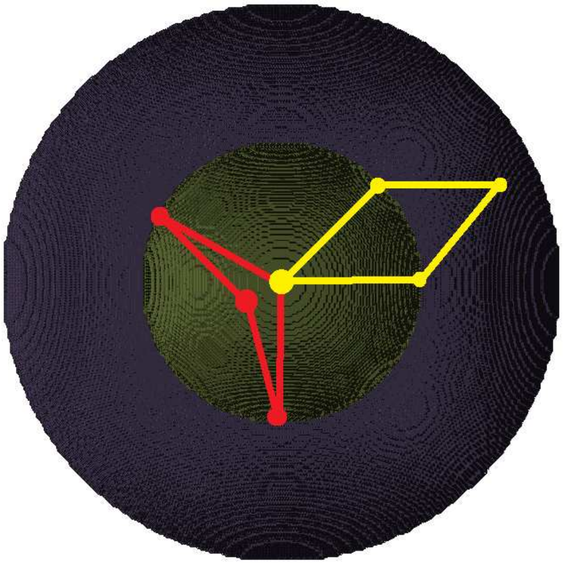


Figure 4.6: A cross section of the reachable volumes of a 4 link closed chain with spherical joints. The first and third joints can reach any point along the inner sphere (green) while the second joint can reach any point inside the outermost sphere (blue). Example configurations are shown in red and yellow.

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 4.10).

4.2.2 Reachable Volumes of Closed Chains

As with other robots, we define the RV-space of a closed chain to be a 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

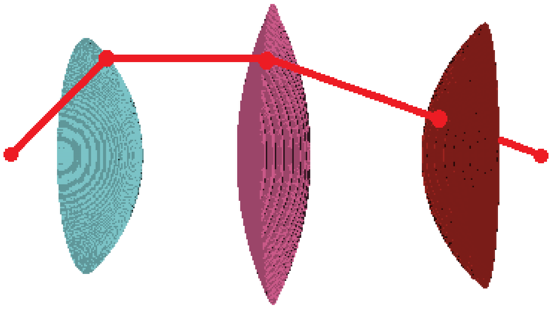


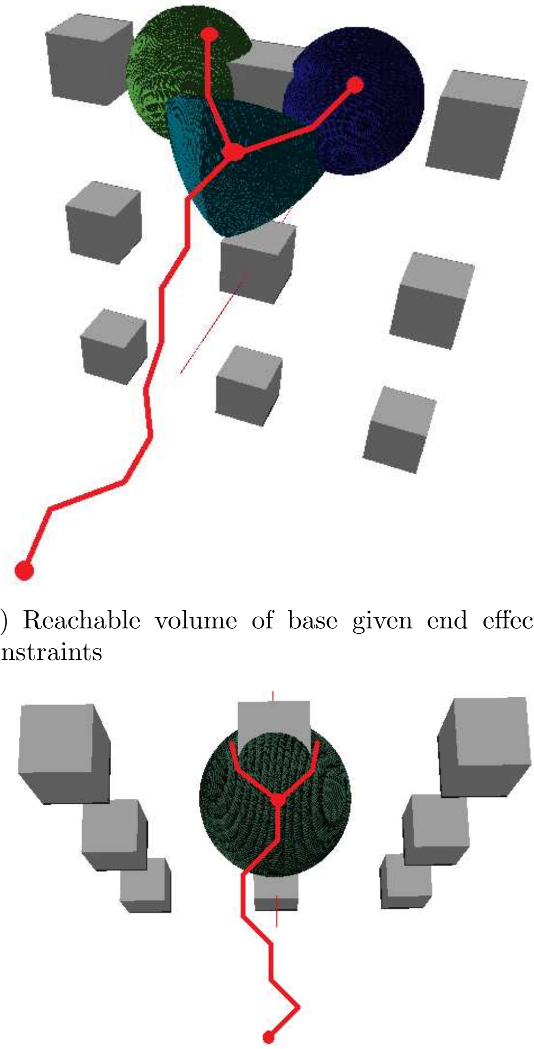
Figure 4.7: The reachable volumes of a 4 link chain with spherical joints of length 9 where the end effector is constrained to a point 8 units away from the base. The first joint must be located in the left-most shell-like region (blue), the second joint must be located in the middle region (pink), and the third joint must be located in the right-most shell-like region (red). An example configuration is shown in red.

chain, each joint is connected to the root by two chains (Figure 4.11(a)) and the reachable volume of the joint is equal to the intersection of the reachable volume of the chains (Figure 4.11(b)). 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.

4.2.3 Complexity of Reachable Volumes in Problems Without Constraints

In this section we study the complexity of reachable volumes. We show that reachable volumes have an $O(1)$ 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 which represents the closet point to the origin that the end effector can reach (0 if it can reach the origin). For single link chains, both



(a) Reachable volume of base given end effector constraints

(b) Reachable volume of end effectors given base constraints

Figure 4.8: (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 base. The reachable volume of the base (teal) given constraints on the end effectors to grasp a cubic object (blue and green). (b) The reachable volume of the end effectors (blueish green) when the base 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.

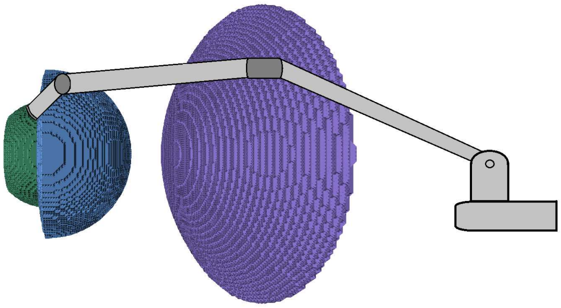


Figure 4.9: The reachable volume of a WAM robot [27] grasping a spherical object. In order to reach the object, the elbow joint must be located in the purple region (right), the second arm joint must be located in the light blue region(center) and the wrist of the grasper must be located in the green region(left). The reachable volumes of the knuckle joints and the object being grasped are not visible because they are contained in the reachable volume of the wrist. This robot has 15 dofs and includes both spherical and planar joints. An example configuration is shown in gray.

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 min value $R1_{min}$ and the max 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 the minimum value $(R1 \bigoplus R2)_{min}$ and the maximum value $(R1 \bigoplus R2)_{max}$ which are computed as follows:

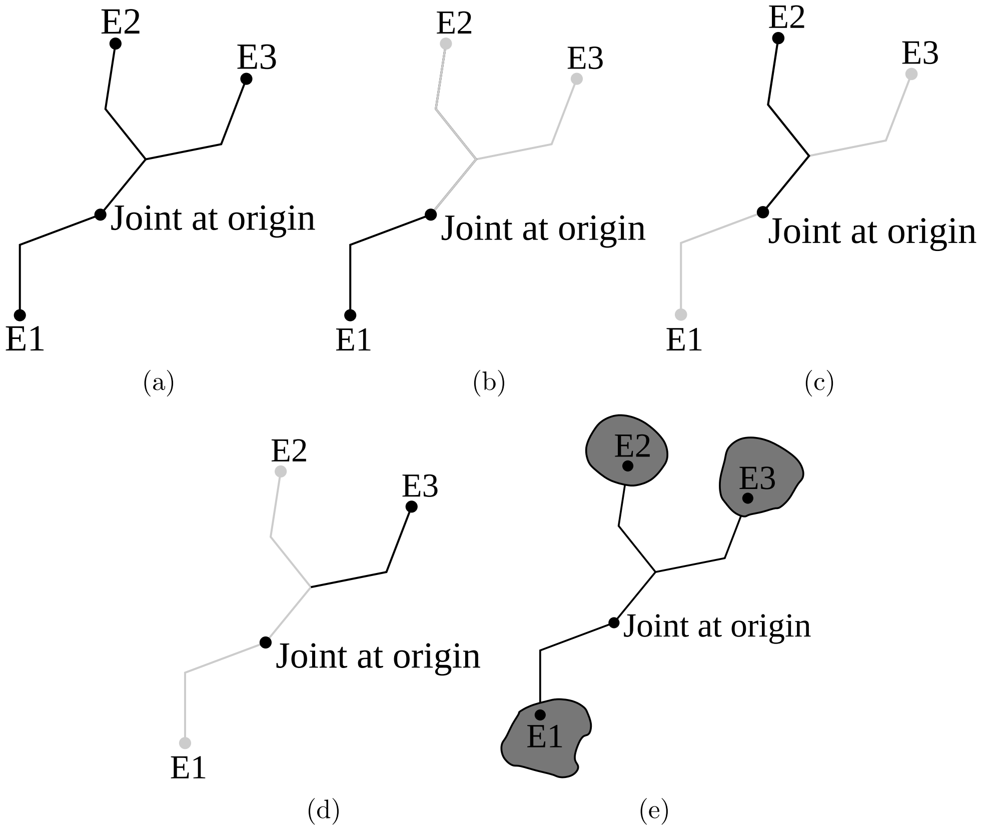
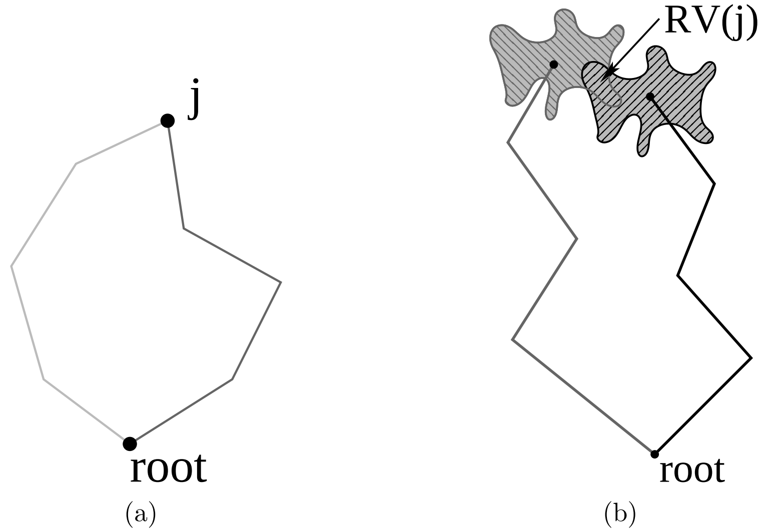


Figure 4.10: For a tree robot with 3 end effectors (E1, E2 and E3) (a) we compute the reachable volumes of each end effector by computing the reachable volumes of the chain connecting it to the origin joint (b–d) resulting in the reachable volume of the end effectors of the linkage (e).

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

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

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



(a) (b)

Figure 4.11: Generating a closed chain configuration: (a) Two open chains connect joint j to root. (b) Reachable volume of j is intersection of reachable volumes of open chains.

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 positions 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 comprised of the links $L_c = \{l_1, l_2, \ldots, l_n\}$, joints $J_c = \{j_0, j_1, \ldots, j_m\}$, and RV-space constraints $S_c = \{S_0, S_2, \ldots, S_m\}$ placed on its joint positions. As a base case, the constrained reachable volume of the chain l_1 is Reachable$(l_1) \cap S_1$. Note that if S_1 is null, then the constrained reachable volume

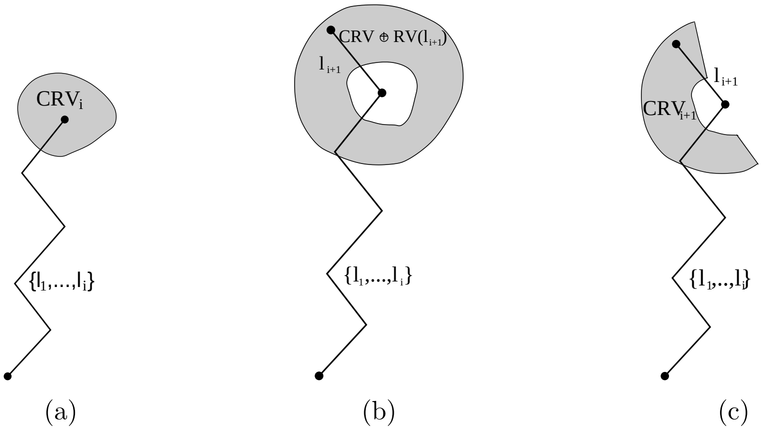


Figure 4.12: (a) CRV_i is the constrained reachable volume of $\{l_1, \ldots, l_i\}$. (b) The region that is reachable by the endpoint of link l_{i+1} is $CRV_i \bigoplus$ ReachableVolume(l_{i+1}). (c) The constrained reachable volume of the chain $\{l_1, \ldots, l_{i+1}\}$ is therefore ($CRV_i \bigoplus$ ReachableVolume(l_{i+1})) $\cap S_{i+1}$.

is the empty set. In this case, no configuration will satisfy the constraint. Now we make the inductive assumption that the constrained reachable volume of the linkage $\{l_1, \ldots, l_i\}$ is CRV_i (see Figure 4.12(a)). In the linkage $\{l_1, \ldots, l_{i+1}\}$, the base of link l_{i+1} coincides with the end of link l_i, and CRV_i is the set of possible locations of this endpoint. The set of points the endpoint of l_{i+1} can reach is therefore $CRV_i \bigoplus$ Reachable(l_{i+1}) (see Figure 4.12(b)), and the set of points that this endpoint can reach while satisfying the constraint S_{i+1} is ($CRV_i \bigoplus$ Reachable(l_{i+1})) $\cap S_{i+1}$ (see Figure 4.12(c)). By induction, the constrained reachable volume of the chain C must be:

$$RV(L_c, S_c) = \begin{cases} RV(l_1) \cap S_1 & |L_c| = 1 \\ (RV(L_c - l_{|L_c|}, S_c - S_{|S_c|}) \bigoplus RV(l_{|L_c|})) \cap S_{|L_c|} & otherwise \end{cases}$$

As with other reachable volumes, the constrained reachable volume does not take into consideration obstacles and it does not exclude configurations with self-collisions.

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. To compute reachable volumes exactly we must be able to compute Minkowski sums and intersections on the geometry. For problems where these computations are not feasible, we compute the reachable volume of the chain without constraints (using the method presented in Chapter 4), 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.

$$RV(C,J,S) = RV_{0,|J|} \cap (S_1 \bigoplus RV_{|J|-1,|J|}) \cap \ldots$$

$$\cap (S_1 \bigoplus RV_{|J|-1,|J|})$$

where $RV_{0,|J|}$ is the reachable volume of the chain and $RV_{j,|J|}$ is the reachable volume of the portion of the chain after joint j (without constraints).

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 $RV_{j,|J|}$ first. Because $RV_{j,|J|}$ is the reachable volume of a chain, its reachable volume is defined by 2 concentric spheres and can be computed as described in the previous paragraph. The Minkowski sum of S_j and $RV_{j,|J|}$ 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 reachable volumes using this method requires time of $O(|J|\ |S|\ C(S))$ time and $O(|S|\ C(S))$ space where $|J|$ is the number of joints, $|S|$ is the number of constraints and $C(S)$ is the complexity of the constraints. Samples can therefore be generated in $O(|J|^2\ |S|\ C(S))$ time (recall that the complexity of reachable volume sampling is linear in the complexity of the reachable volumes).

4.4 Computing the Reachable Volumes of All Joints

In Section 4.1.2 we presented a method for computing the reachable volume of a single joint (Algorithm 1). This is ideal for an application such as sampling where you only need to know the reachable volume of a single joint at any given time (i.e., the joint you are are sampling). However for applications such as robot design, control and modeling, you may need to know the reachable volume of all of the joints in the robot. You could apply the method presented in Section 4.1.2 to every joint in the robot, but that would be very inefficient. In this section we present an efficient method for computing the reachable volume of all of the joints in the robot. We show that the running time of this method is $\mathcal{O}(|J|$*diameter$(R))$ where $|J|$ is the number of joints in the robot and diameter(R) is the diameter of the robot. In comparison, computing the reachable volume of each joint separately results in a slower running time of $\mathcal{O}(|J|^2)$.

Algorithm 2 is a dynamic programming method that initializes the reachable volume of each joint to be the constraints for that joint. It then iteratively updates the reachable volume of each joint to be the intersection of its reachable volume and the set of points it can occupy given the reachable volume of its neighbors. Note that the set of points a joint can occupy given the reachable volume of its neighbors is equal to the Minkowski sum of the reachable volume of the neighbor and the reachable volume of the link connecting the joint to the neighbor (see Section 4). If the reachable volumes of all joints are not changed over an iteration, the algorithm stops.

In order to prove the correctness of Algorithm 2, we present the following lemmas:

Lemma 2. *For all $j \in J$, the reachable volume of j will always be a subset of RV_j during all iterations of Algorithm 2.*

Algorithm 2 Computing the reachable set of all joints in the robot

```
Input: A robot $R$ that contains no cycles
Output: The reachable set of all joints in $R$
$RV_0 = (0, 0, 0)$
for all $v_j \in R\setminus$ root do
    $RV_{j,0} = S_j$
Changed = true, $i = 0$
while Changed do
    Changed = false
    for all $v_j \in R\setminus$ root do
        $RV_{j,i+1} = \bigcap_{j' \in Neighbors(j)} RV_{j',i} \bigoplus RV(\text{link}(j, j')) \cap S_j$
        if $RV_{j,i} \neq RV_{j,i+1}$ then
            Changed=true
    $i$++
return $RV_i$
```

Proof. We show this using induction. As a base case, $RV_{0,0}$ is initialized to (0,0,0) because the root is at the origin in RV-space, while the reachable set $RV_{j,0}$ for all other joints j is initialized to the joint's constraints S_j. Because a joint cannot be located outside of its constraint, the reachable set of j must be in $RV_{j,0}$.

Assume that for all j the reachable volumes of j are a subset of $RV_{j,i}$. During iteration $i + 1$ this algorithm sets

$$RV_{j,i+1} = \bigcap_{j' \in Neighbors(j)} RV_{j',i} \bigoplus RV(\text{link}(j, j')) \cap S_j \text{ for all joints } j.$$

By our inductive assumption, $RV_{j',i}$ must include the reachable set of j' for all $j' \in Neighbors(j)$.

$$\bigcap_{j' \in Neighbors(j)} RV_{j',i} \bigoplus RV(\text{link}(j, j')) \cap S_j$$

must therefore include the reachable set of j. By induction, $RV_{j,i}$ must always include the reachable set of j. □

Lemma 3. *Let $R'_{j,i} \subseteq R$ be the subset of R that is within i hops of j. After the ith iteration, $RV_{j,i}$ will be a subset of the reachable volume of vertex j in $R'_{j,i}$.*

Proof. We show this using induction. As a base case, $RV_{j,0}$ is set to S_j which is the reachable volume of j in $R'_{j,0}$ and consists only of the vertex j.

Assume that for all $j \in J$, $RV_{j,i}$ is a subset of the reachable volume of j in $R'_{j,i}$. During iteration $i+1$, Algorithm 2 sets

$$RV_{j,i+1} = \bigcap_{j' \in Neighbors(j)} RV_{j',i} \bigoplus RV(\text{link}(j,j')) \cap S_j$$

for all $j \in J$. We first note that the graph $R'_{j,i+1}$ must be a tree (because it is a connected subset of R which had no cycles). As a convention, we define j to be the root of $R'_{j,i+1}$. Let j' be an arbitrary neighbor of j in $R'_{j,i+1}$. j' and all of its descendants are in $R_{j',i}$, so by our inductive assumption $RV_{j,i}$ must be a subset of the region reachable by j' under the constraints of j' and its descendants in $R'_{j,i+1}$. $RV_{j',i} \bigoplus RV(\text{link}(j,j'))$ must therefore be a subset of the region that j' can reach while satisfying the constraints of j' and its descendants. $\bigcap_{j' \in Neighbors(j)} RV_{j',i} \bigoplus RV(\text{link}(j,j')) \cap S_j$ must therefore be a subset of the region that j can occupy while satisfying its own constraint and the constraints of all other joints in $R'_{j,i+1}$. □

Corollary 2. *After at most diameter(R)+1 iterations, Algorithm 2 will return the reachable volume of all vertices.*

Proof. First we observe that the reachable volume of a joint j cannot contain any points that are not in S_j, otherwise the constraint S_j would be violated. We also observe that if two joints j and j' are connected by an edge $e(j,j')$, then every point p in the reachable volume of j must be length(e) away from a point in the reachable volume of j', otherwise it would be impossible to place j at p without placing j' outside of this reachable volume. From these observations we conclude that if at i, RV_i is equal to the reachable volumes of R, then for all $j \in J$, $RV_{i,j} = \bigcap_{j' \in Neighbors(j)} RV_{j',i} \bigoplus RV(\text{link}(j, j')) \cap S_j$. This means that $RV_{i,j} = RV_{i+1,j}$ for all j and that the algorithm will return RV_{i+1} (which is equal to RV_i) on the next iteration.

For all $j \in J$, $R'_{j,diameter(R)} = R$, so by Lemma 2 $RV_{j,diameter(R)}$ must be a subset of the reachable volume of j in R. In Lemma 2 we showed that for all i the reachable volume of j will be a subset of $RV_{j,i}$, which means that $RV_{v,diameter(R)}$ must be equivalent to the reachable volume of j. Because $RV_{diameter(R)}$ contains the reachable volume of all the joints in R, we know that Algorithm 2 will return this reachable volume on the next iteration. Now consider the case where Algorithm 2 returns on some arbitrary $i \leq diameter(R)+1$. In order for it to return, RV_{i-1} must be equal to RV_i. If this algorithm were to continue running, then inductively all $RV_{i'>i}$ must be equal to RV_i. RV_i must therefore be equal to $RV_{diameter(i)+1}$, which is the reachable volumes of R. □

5. SAMPLING BASED MOTION PLANNING WITH REACHABLE VOLUMES*

In this chapter we show how sampling based motion planning can be used with reachable volumes. We first present reachable volume 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 Sampling with Reachable Volumes

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 chains, tree-like robots, and closed chains with constraints.

5.1.1 Generating Configurations for Chains

To generate samples of a chain robot without constraints, we first compute the reachable volume of the end effector of the chain (Algorithm 1). Here we select the midpoint joint as the splitting joint on line 3. For problems requiring multiple samples, this computation can be performed once as a prepossessing step. We then recursively position the internal joints of the chain by selecting a joint from the chain, "breaking" the chain at this joint, and computing the reachable volumes of both pieces of the chain (see Figure 5.1). We then translate the reachable volume of the second chain so that the base of the second chain is located at the origin and compute the intersection of these reachable volumes. This intersection is the region

*Reprinted with permission from "Sampling based motion planning with reachable volumes: Theoretical foundations" by Troy McMahon, Shawna Thomas, and Nancy M. Amato, 2014. Proc. IEEE Int. Conf. Robot. Autom. (ICRA), pages 6514-6521, Copyright 2014 by IEEE.

of second chain's RV-space that can contain the selected joint. We then randomly select a position for the joint from this intersection. This process is described in Algorithms 3 and 4. We convert this sampled point to a point in the RV-space of the robot by adding the position of the base of the left chain. Finally, we recurse on the subchains formed by breaking the chain at the newly sampled joint.

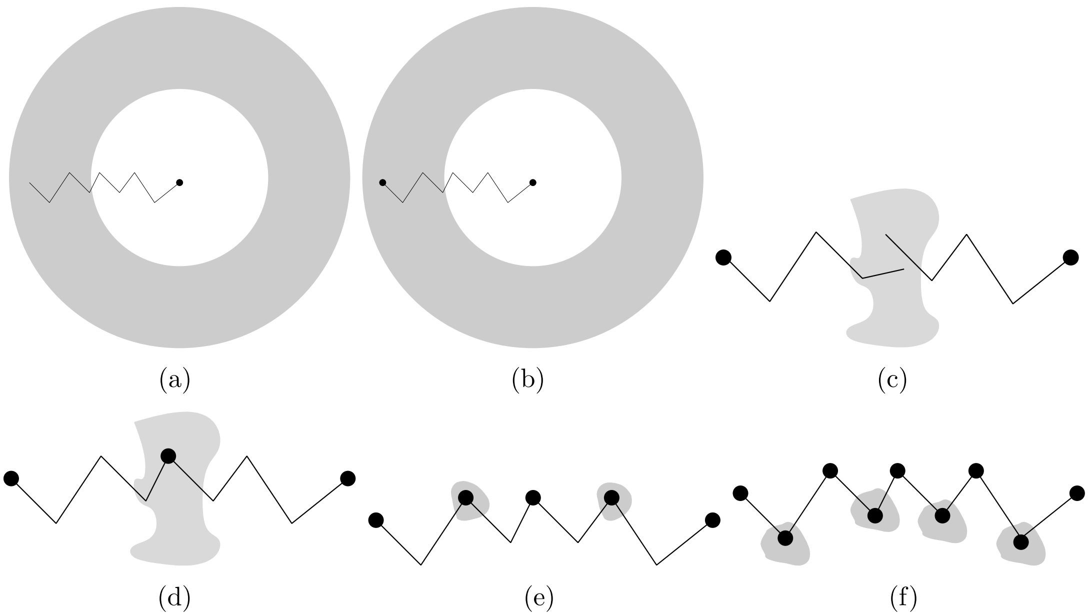


Figure 5.1: Generating a configuration for a chain robot: (a) Compute the reachable volume of the chain. (b) Set the position of the end effector of the chain to be a point from this volume. (c) Bisect the chain and compute intersection of the reachable volumes of the two pieces. (d) Set the midpoint of the bisected chain to be a point from the intersection of these reachable volumes. (e,f) Continue until all joints are placed.

For planar and prismatic joints, we parse 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

Algorithm 3 Generating configurations for chains

```
Input: A chain $C$
Output: A randomly sampled configuration of $C$ by setting values for its dof
Compute the reachable volume $RV_C$ of $C$
Set the end effector of $C$ to a be random point from $RV_C$
SampleInternal($C$)
Convert Sample to C-Space Sample
Randomly sample translational and rotational coordinates
```

Algorithm 4 SampleInternal

```
Input: A chain $C$ whose end effectors have already been sampled/set
Output: A randomly sampled configuration of $C$ in RV-space by setting values for its dof
if $C$ only has 1 link then
    return
Let $j$ be the joint at the midpoint of $C$
Let $C_l$ be the portion of $C$ to the left of $j$
Let $C_r$ be the portion of $C$ to the right of $j$
$RV_l$ = reachable volume of $C_l$
$RV_r$ = reachable volume of $C_r$
The position of $j$ = random point from $RV_l \cap (RV_r + \text{base}_r - \text{base}_l)$ + position of the base of $C_l$ in RV-space of the robot
SampleInternal($C_l$)
SampleInternal($C_r$)
```

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 joints reachable volume.

We next show that the running time of the reachable volume sampler is linear 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 operation (where $(|L|)$ is the number of links in the robot). In Section 4.2.3 we show that the complexity of computing the Minkowski sums of two reachable volumes is proportional to the complexity of the 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 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 j_l through j_r. When computing the reachable volume of the chain in the first step of Algorithm 4, we recursively compute the reachable volumes of the chain j_l through j (which we will denote as $RV_{j_l,j}$) and j through j_r (RV_{j,j_r}). We next recall that reachable volumes are symmetric which means that the reachable volume of the chain j_r through j ($RV_{j_r,j}$) is equal to RV_{j,j_r}. Algorithm 4 samples j by placing j in the intersection of $RV_{j_l,j}$ and $RV_{j_r,j}$. Because $RV_{j_l,j}$ and $RV_{j_r,j}$ were computed while computing the reachable volume of the chain, we don't need to compute them during sampling. We only need to translate $RV_{j_l,j}$ by j_l and translate $RV_{j_r,j}$ by j_r, which is done by defining the bases of $RV_{j_l,j}$ and $RV_{j_r,j}$ to be at j_l and j_r (which can be done in constant time). Joint j is then sampled by selecting a position from the intersection of $RV_{j_l,j}$ and $RV_{j_r,j}$ as described in Section 5.1.3.

Samples are generated 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 $RV_{j_r,j}$. Testing if a point is in a reachable volume is equivalent to testing if the distance between the point and the base of the chain (i.e., j_l or j_r) is between the minimum and maximum values for that chain (as described in Section 5.1.3), 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 position (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 to a joint angle sample which can be done in $\mathcal{O}(|L|)$ time as described in Section 5.1.1. 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 Algorithm 5 and Figure 5.2) which is done by dividing the tree at any joint with more than two neighbors. The order in which we partition the robot should not have any effect on the computation time or the probability distribution of the sample, so we select the order in which the tree is partitioned at random. We then use use Algorithm 3 to generate a reachable volume configuration for each chain. We translate each reachable volume configuration into the RV-space of the root of the robot which is done by translating the configuration of each chain by the position of the first joint in the chain in the RV-space of the root of the robot. We then convert this into a C-space configuration and randomly sample any rotational and translational coordinates.

Algorithm 3 samples each branch in linear time with respect to the size of the

Algorithm 5 Generating configurations for tree-like robots

Input: A tree-like robot T

Output: A randomly sampled configuration of T

```
1: Decompose $T$ into chains
2: Generate configurations for each of these chains using Algorithm 3
3: Concatenate these configurations to form a configuration of $T$
```

branch, so the total time to sample all branches 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 $\mathcal{O}(|L|)$.

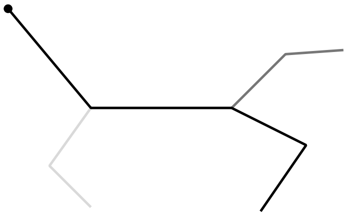


Figure 5.2: We generate a configuration for a tree-like robot by applying our method to each of the branches (black, dark gray, light gray).

Generating configurations for closed chains: To compute configurations for closed chains, we decompose the closed chain into two open chains. We observe that if the two open chains are in configurations that share the same endpoints, then they can be combined to form a configuration of the closed chain. We now note that in RV-space both chains are rooted at the origin (we assume that the same endpoint is rooted for both chains). In order for both chains to reach the same endpoint in workspace, they must reach the same point in RV-space (see Chapter 4). The set of possible positions for the end effectors of the two chains is therefore the intersection of the reachable volumes of the two chains. We can therefore select a point from this intersection to be the endpoint of the two chains (see Algorithm 6 and Figure 5.3) and then sample the other points of the chains as described in Section 5.1.1. As with chains and tree-like robots, this method yields an RV-space configuration that can be transformed to a C-space configuration by setting the translational and rotational coordinates of the robot.

The reachable volume of each branch is found by computing the Minkowski sums of the links in the branch 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 branch 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.

Algorithm 6 Generating configurations for single loop closed chains

```
Input: A single loop closed chain $C$
Output: A randomly sampled configuration of $C$
Let $root$ be an arbitrary joint from $C$
Let $j$ be another arbitrary joint from $C$ such that $j \neq root$
Let $C_1$ and $C_2$ be the chains formed by breaking the closed chain at $j$ and $root$
Compute ReachableVolumes($C_1$) and ReachableVolume($C_2$) using Algorithm 1
Place $j$ in ReachableVolume($C_1$) $\cap$ ReachableVolume($C_2$)
SampleInternal($C_1$)
SampleInternal($C_2$)
Convert sample to a C-Space sample
Randomly sample translational and rotational coordinates
```

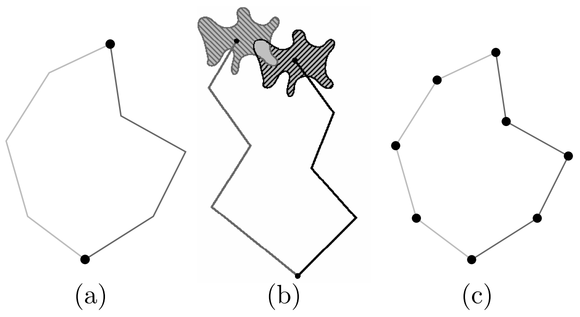


Figure 5.3: Generating a closed chain configuration: (a) Break the closed chain into two open chains. (b) Compute the reachable volumes of the two chains (striped regions). (c) Randomly select a point from the intersection of their reachable volumes and use Algorithm 3 to sample the positions of the internal joints.

5.1.3 Sampling in the Intersection of Reachable Volumes

We present a set of methods for computing samples in reachable volumes and discuss when each method is applicable. These methods can be used by the algorithms in Sections 5.1 when sampling joint positions.

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 allied bounding box around the reachable volume 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 this is in all of the reachable volumes. This method can be applied to situations in which none of the other methods is 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. Additionally, 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 that is optimal 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 ability to reuse the reachable volumes $RV_{jl,j}$ and $RV_{jr,j}$ that were computed while computing the reachable volume of the linkage during the sampling step. Unfortunately, reachable volumes for problems with constraints are not symmetric. This means that one cannot use the reachable volume $RV_{jr,j}$ that is computed during the initial step of the algorithm to obtain the reachable volume $RV_{j,jr}$ that is needed during sampling. In order to use the methods described in Sections 5.1.1 and 5.1.2 for constrained problems, it would be necessary to compute $RV_{j,jr}$ during each level of sampling. This computation requires $\mathcal{O}(|L|)$ Minkowski sum operations and will result in a running time that is $\mathcal{O}(|L|log(|L|))$ in the complexity of these operations.

Algorithm 7 shows how to use reachable volumes to compute samples for problems with constraints. It first sets the position of the root of the robot to be (0,0,0) which is its location in RV-space by definition. It then selects an end effector j and calls the function ComputePartialRV (Algorithm 8) to set RV_j to be the reachable volume of $j \in J$ in the subset of the robot comprised of the children of j in the traversal. It then calls the function ComputeSampleHelper (Algorithm 9) that performs a second depth first traversal of this tree. During this traversal, the position of every joint j is set to be a random point in the intersection of RV_j and $P_{jprv} \bigoplus$ ReachableVolume(edge(j,parent(j)), which is the volume of space that j can occupy given the placement of its parent in the traversal. The reachable volumes are computed as follows:

- If parent(j) is a spherical joint, then ReachableVolume(edge(j,parent(j)) is a sphere centered at j with a radius equal to the the length of edge(j,parent(j).
- If parent(j) is a planar joint, then ReachableVolume(edge(j,parent(j)) is a circle in the plane of the joint with a radius equal to the the length of edge(j,parent(j)).
- If parent(j) is a prismatic joint, then ReachableVolume(edge(j,parent(j)) is the line segment defined by the points d_{min} and d_{max} (see Section 3.1).

After all of the joints have been sampled, Algorithm 7 transforms the resulting RV-space sample into a joint angle configuration (as discussed in Section 4.1.1) and randomly samples the position and orientation of the robot to form a C-space sample. The resulting C-space sample can be used by sampling-based motion planners like PRMs.

These algorithms perform two depth first traversals of the robot. During the first traversal we compute the partial reachable volume of each joint from the partial reachable volumes of its children in the traversal which requires one Minkowski sum

Algorithm 7 Compute sample for chains or tree-like robot which satisfies constraints S

```
Input: A robot $R = (J, E)$ that contains no cycles, $root$ = an end effector of $R$, a set of constraints $S$ on $J$
Output: A reachable volume configuration $P$ that satisfies $S$
$P_{root} = (0,0,0)$
Let $j$ be an arbitrary end effector that $\neq root$
$RV_j$ = ComputePartialRV($j, \emptyset$,array($|J|$))
Let $P_j$ be a random point from $RV_j$
$RVSample$ = ComputeSampleHelper($j, \emptyset, P_j, RV_j$)
$c$ = CSpaceSample(RandomPosition, RandomOrientation, JointAngles($RVSample$))
return $c$
```

Algorithm 8 ComputePartialRV

```
Input: A robot $R = (J, E)$, a set of constraints $S$ on $J$, and the subset $P$ of $J$ that have been assigned positions
Output: The reachable volume $RV_j$ of $j$ in the robot $R\backslash$ branch($j_{prv}$) under the constraints $S$ and the partial positioning $P$
ComputePartialRV($j, j_{prv}, RV$)
if $j \in P$ then
    return Position($j$)
$RV_j = S_j$
for all $j' \in$ Neighbors($j$) $\backslash j_{prv}$ do
    $RV_{j'}$ = ComputePartialRV($j', j, RV$)
    $RV_j = RV_j \cap RV_{j'}$
return $RV$
```

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 reachable volume of the link connecting the joint to its parent, translate that reachable volume by the position of the parent, compute the intersection of this reachable volume and the partial reachable volume of the joint (computed in the first traversal), and set the position of the joint to be a random point from this intersection. This method

Algorithm 9 ComputeSampleHelper

```
Input: Joints $j$ and $j_{prv}$, a set of partial positions $P$, and a reachable volume $RV$
Output: An updated set of partial positions $P$ including $j$
if $j \in P$ then
    return $P$
else
    if $RV_j \cap (P_{jprv} \bigoplus \text{ReachableVolume}(\text{edge}(j, j_{prv}))) = \emptyset$ then
        return Fail
    $P_j$ = random point from $RV_j \cap (P_{jprv} \bigoplus \text{ReachableVolume}(\text{edge}(j, j_{prv})))$
for all $j' \in \text{Neighbors}(j) \setminus j_{prv}$ do
    ComputeSampleHelper($j', j, P, RV$)
return $P$
```

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

We next discuss how to use reachable volumes to generate samples for closed chain robots with constraints(Algorithm 10).

A single loop closed chain robot is any robot of genus 2. We first define the root of this robot to be one of the joints along the closed chain of the robot (such that if the root were removed the robot would contain no cycles). We then select one of the root's neighbors j that is also located on the closed chain. Because the root is at the origin, edge($root, j$) imposes the constraint that j be the length of edge($root, j$)) away from the origin. We can therefore remove edge($root, j$) and replace it with a constraint that j be within ReachableVolume(edge($root, j$)) of the origin which is done by setting S_j to be $S_j \cap$ ReachableVolume(edge($root, j$)). Recall that if there is no constraint on j, then S_j is the entire workspace. Once the edge has been removed, the robot is a tree rooted at $root$. We then sample each of the branches of this tree in the same manner that we generate samples for a tree-like robot in Algorithm 7. We convert this sample to a joint angle configuration and randomly sample the translational and rotational coordinates of the robot. This algorithm performs two depth first traversals on each branch of the robot. As with Algorithm 7, this method preforms $\mathcal{O}(|L|log(|L|))$ Minkowski sum operations and $\mathcal{O}(|L|log(|L|))$ intersection operations, so its running time is also $\mathcal{O}(|L|log(|L|))$ in the complexity of these operations.

Algorithm 10 Compute sample for closed chain robots which satisfies constraints S

Input: A robot $R = (J, E)$ that contains a single closed chain, $root$ = a joint on the closed chain in R, and a set of constraints S on J.

Output: A configuration that satisfies S

```
$P_{root}$ = (0,0,0)
Let $j$ be an arbitrary joint from Neighbors($root$)
$S_j = S_j \cap$ ReachableVolume(edge($root, j$))
Remove edge($root, j$) from $R$
for all $j' \in$ Neighbors($root$) $\setminus j$ do
    Let $j$ be an end effector from the branch composed of $j'$ and its descendants
    $RV_j$=ComputePartialRV($j, root$, array($|J|$))
    Let $P_j$ be a random point from $RV_j$
    $RVSample$ = ComputeSampleHelper($j, \emptyset, P, RV$)
$c$ = CSpaceSample(RandomPosition, RandomOrientation, JointAngles($RVSample$))
return $c$
```

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 (Figure 5.4 and Algorithms 11, 12). This stepping function will serve as a primitive operation for a reachable volume local planner (Section 5.3) and a reachable volume RRT [31]. This method starts with an initial configuration q, a specified joint j, and a target position v. It perturbs q by moving j by δ in the direction of v (Figures 5.4(b) and 5.4(c)). It then updates the position of j and its descendants to ensure that all joints are in the reachable volume of their parents which will ensure that the joint positions defined the new sample q' correspond to a constraint-satisfying configuration (Figures 5.4(d) and 5.4(e)).

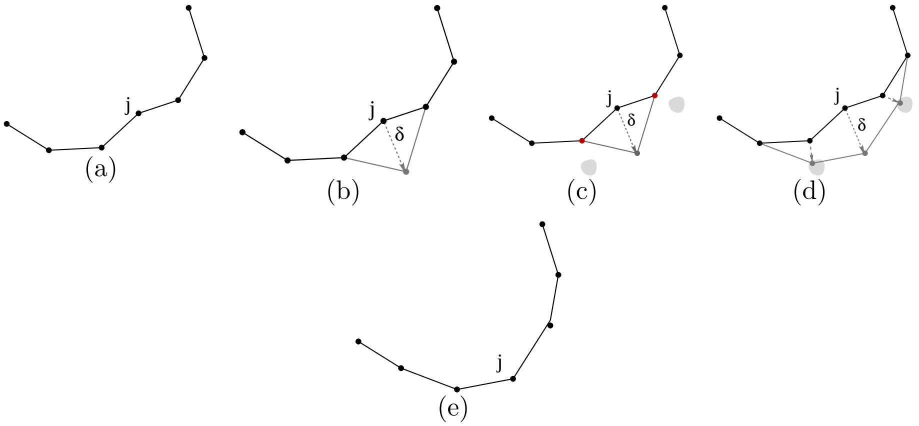


Figure 5.4: Reachable Volume Stepping: We step one joint j by a distance of δ then update the j's descendants to be in their reachable volumes given j's new position. The gray regions are the reachable volumes of the third and fifth joints after j is stepped. These joints are repositioned to be in their reachable volumes (d) resulting in a configuration in which all joints are in their reachable volumes (e).

We observe the following about perturbing a joint:

Observation 4. *If we perturb 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 position constraints in the* *environment, then the final configuration must also satisfy all joint position 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 position of the joint. □

Algorithm 11 Reachable Volume Stepping

```
Function: RV-Step($q$, $j$, $p_{target}$, $\delta$)
Input: A cfg $q$, a joint $j$ and a target position $p_{target}$, a stepping parameter $\delta$
Output: A cfg in which the joint $j$ has been perturbed by $\delta$ in the direction of $p_{target}$
let $p_{init}$ = position of $j$ in $q$
$p_{new}$=$p_{init}$+($p_{target}$-$p$)*$\delta$
if $p_{new} \in$ RV($j$.LeftParent) $\cap$ RV($j$.RightParent)
  $q_{new}$=copy($q$)
  Set position of joint $j$ to be $p_{new}$ in $q_{new}$
  Reposition($q_{new}$,$j$.LeftChild)
  Reposition($q_{new}$,$j$.RightChild)
  return $q_{new}$
return NULL
```

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 if the descendants of j are still within the reachable volume of their parents. To do this we recursively test the descendants of j (Algorithm 12). 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 position that is in the intersection of the reachable volumes of the joint's parents and near the original position of the joint. When repositioning a joint we know that all previously repositioned joints were

Algorithm 12 Method for repositioning descendants

```
Function: Reposition($q$,$j$)
Input: A cfg $q$ and a joint $j$
Output: A cfg with all $j' \in j \cap$ descendants($j$) in the intersection of the reachable
    volume of their parents
if $j \in$ RV($j$.LeftParent) $\cap$ RV($j$.RightParent)
  return $q$
Adjust position of $j$ in $q$ to be within RV($j$.LeftParent) $\cap$ RV($j$.RightParent)
if $j$.LeftChild $\neq$ NULL
  Reposition($q$,$j$.LeftChild)
if $j$.RightChild $\neq$ NULL
  Reposition($q$,$j$.RightChild)
  return $q$
```

placed in their reachable volumes, so there must exist a sample for the positioning. 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 a reachable volume configuration in which all of the joints are located in the intersection of the reachable volumes of their parents. Such a configuration must correspond to a feasible positioning 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 nodes are affected by a 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 robots 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 [30]). 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 structure of robot:** 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 Chapter 8, this planner can be used by PRMs to find local paths that satisfy a problem's constraints.

The reachable volume local planner (Algorithm 13) connects two configurations, c_1 and c_2, by using reachable volume stepping to move each joint to its position in the second configuration. To accomplish this, it performs a traversal of the joints in the reachable volume data structure (see Section 5.2). During each iteration of the traversal, it uses reachable volume stepping to move the joint from its position in c_1 to its position in c_2.

Algorithm 13 Reachable Volume Local Planner

```
Input: Cfgs $c_1$ and $c_2$, a step size $\delta$
Output: Boolean value indicating if a path was found
queue.push_back($j_{root}$)
while $j$=queue.pop_front() do
    $c'$=$c_1$
    while position of $j$ in $c' \neq$ position of $j$ in $c_2$ do
        $p_{target}$=position of joint $j$ in $c_2$
        $c'$ = RV-Step($c_1$, $j$, $p_{target}$, $\delta$)
        if $c'$==NULL OR invalid($c'$) then
            return $false$
    queue.push_back(children($j$))
$success$ = RigidBodyLocalPlanner($c'$,$c_2$)
return $success$
```

Figure 5.5 is an example of the reachable volume local planner (with a binary reachable volume ordering) being applied to a 5 link chain. The local planner first steps the end effector of the robot from its position in c_1 to its position in c_2 (5.5(a)). It then steps the third joint (5.5(b)), then the fourth joint (5.5(c)) to their positions in c_2. This results in the configuration c_2 (5.5(d)). Note that we are always stepping parents in the reachable volume data structure before their children to ensure that stepping a node will not change the position of any nodes that have already been stepped.

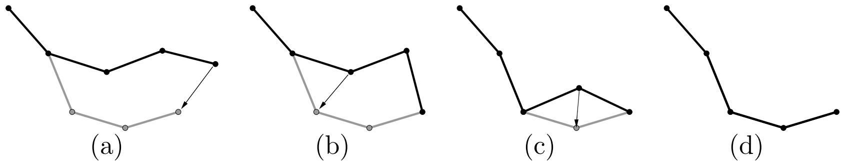


Figure 5.5: The reachable volume local planner uses reachable volume stepping to move the joints of a linkage from their positions in one configuration (black) to their positions in a second (gray).

The sequence of steps covered by the local planner forms a path from c_1 to c_2. 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 configurations of c_1 and c_2 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 configuration, and perform the rest of the rigid body transformation. This is analogous to the rotate-at-S local planner [1].

5.4 Reachable Volume Distance Metric

We define a reachable volume distance metric that measures the distance traversed during reachable volume stepping. The reachable volume expand function and local planner construct paths by moving each of the joints from its position in the first configuration to its position in the second configuration, so a good approximation would be the sum of the distances between the joints in reachable volume space (Figure 5.6). 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):

$$D_{tran+rv}(c_1, c_2) = s * \text{Euclidean}(Base_{c1}, Base_{c2}))$$

$$+(1-s) * \sum_{j \in J} \text{Euclidean}(j_{c1}, j_{c2})$$

where j_{c1} and j_{c2} are the position of j in c_1 and c_2 in RV-space, $Base_{c1}$ and $Base_{c2}$ are the position and orientations of the base in c_1 and c_2, and s is a scaling factor.

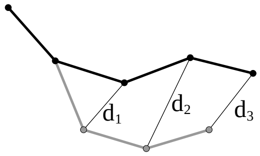


Figure 5.6: The reachable volume distance between two samples (black and gray) is the sum of the distances between the joints of the configurations in reachable volume space. Here the reachable volume distance is $d_1 + d_2 + d_3$.

6. REACHABLE VOLUME PRM

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

6.1 Method Overview

The reachable volume PRM uses the reachable volume sampler, local planner and distance metric in the PRM framework presented in [21]. The reachable volume PRM (Algorithm 14) constructs a roadmap (N,E) by iteratively generating samples, c, using reachable volume sampling. If c is valid, the method adds it to the roadmap, and then attempts to connect it to the k closest nodes $n \in N$ using the reachable volume local planner. If a connection is found, then it adds an edge from n to c to the roadmap.

Algorithm 14 Reachable Volume PRM

```
Input: A motion planning problem, an integer $k$
Output: A roadmap ($N$,$E$)
$N = \emptyset$
$E = \emptyset$
loop
    $c$ = Reachable Volume Sample
    if $valid(c)$ then
        $N = N \cup c$
        $N_c = k$ closest neighbors to $c$ by reachable volume distance
        for all $n \in N_c$ do
            if Reachable volume local planner finds path from $c$ to $n$ then
                $E = E \cup edge(c, n)$
```

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, while 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 sample space, then it will be unable to solve problems that require samples from regions it doesn't cover, no mater 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 over-sample 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 has a sufficient probability of generating samples over a problem's entire samples space, and especially important that it generates samples in the critical regions of an environment. Unlike most methods which 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.

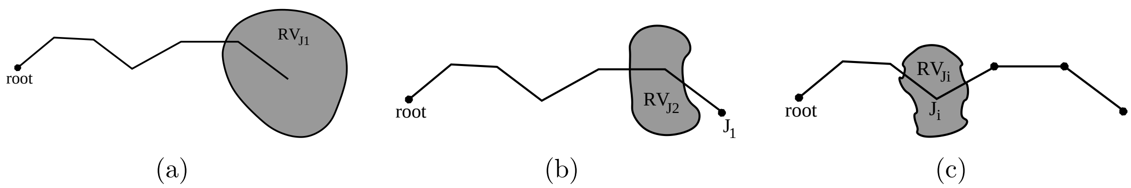
6.2.2 Distribution of Reachable Volume Samples

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. RV-space configurations consist of the position of each of the joints of the robot in RV-space. The distribution of RV-space samples is therefore the distribution of these joint positions. 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 sample and the reachable volume of each subsequent joint is uniform across its reachable volume given the position of all of the joints that have already been sampled (see Figure 6.1).

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

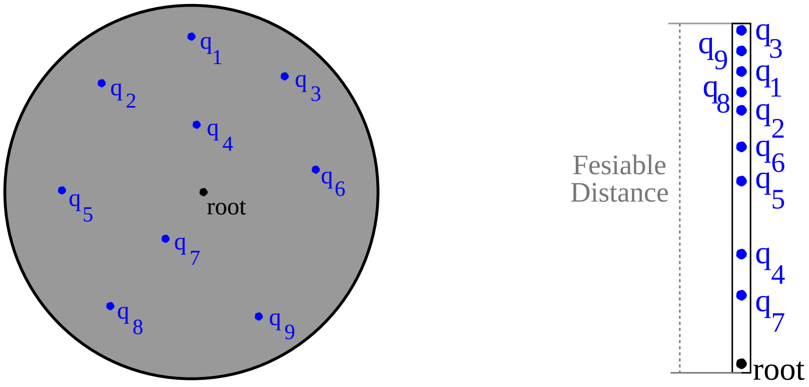


(a) (b) (c)

Figure 6.1: 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 position 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 position 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.

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 6.2). The number of samples within a distance d of the root 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. (see Figure 6.2). 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 is skewed towards the lower portion of their distance range.

Also 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



(a)

(b)

Figure 6.2: 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).

further from the root. For unconstrained robots them method will produce a similar sample distribution to uniform sampling.

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 position of the joints that are already placed.*

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 position 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 possibly 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. REACHABLE VOLUME RRT (RVRRT)*

We introduce an RRT expansion strategy called RV-Expand (Algorithm 15) that uses reachable volume stepping to generate RRT nodes. This strategy takes as input a random sample, q_{ran}, and its nearest neighbor in the graph, q_{near}. It then steps one of the joints in q_{near} by a distance δ in the direction of the position of the joint in q_{ran} to form a candidate sample, q_{new}. Because RV-space encodes the relative joint positions of the robot, stepping a reachable volume sample will change the relative position of the joints and thus alter the internal coordinates of the robot. For free-base robots we also step the translational and rotational coordinates in the direction of q_{ran}.

```
Algorithm 15 RV-Expand
Function: RV-Expand($q_{ran}$, $q_{near}$, $\delta$, $s$)
Input: A cfg $q_{ran}$, its nearest neighbor $q_{near}$, a stepping parameter $\delta$ and a distance
    metric scaling factor $s$
Output: A new cfg to be added to the RRT
if free base then
    $\delta_{tran}$= $\delta * s* \frac{Distance(q_{ran},q_{near})}{Distance_{tran+rv}(q_{ran},q_{near})}$
    $\delta$ =$\delta$ - $\delta_{tran}$
Select a joint $j$ to perturb
$q_{new}$ = RV-Step($q_{near}$,$j$,position of $j$ in $q_{ran}$, $\delta$)
if free base then
    Step rotational and translational dofs by $\delta_{tran}$ in direction of $q_{ran}$
return $q_{new}$
```

We make no assumptions about how q_{ran} is generated (e.g., by uniform sampling

*Reprinted with permission from “Reachable volume RRT.” by Troy McMahon, Shawna Thomas, and Nancy M. Amato, 2015. Proc. IEEE Int. Conf. Robot. Autom. (ICRA), pages 2977-2984, Copyright 2015 by IEEE.

as in [24] or by reachable volume sampling [30]). Here, we use reachable volume sampling [30].

RV-Expand may select a joint to perturb (line 4 of Algorithm 15) in a variety of ways:

- **Random:** select a joint at random. This is advantageous because it requires no overhead and it ensures that all joints have a chance of being selected.
- **Most Distant:** to select the joint that is furthest from its counterpart in the random sample, q_{ran}.
- **Probabilistic:** assign each joint a selection probability. A joint's selection probability could be proportional to the distance between it and its counterpart in q_{ran}.

8. EVALUATION OF REACHABLE VOLUMES*

In this chapter 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 8.1 we discuss our experimental setup and introduce the environments used. In Section 8.2 we evaluate reachable volume sampling, in Section 8.3 we evaluate the reachable volume local planner and distance metric, and in Section 8.4 we evaluate the quality of roadmaps produced using reachable volume techniques.

8.1 Experimental Setup

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

8.1.1 Methodology

When designing our experiment set we took into consideration the experiment sets in papers that address similar problems (particularly [47, 48, 19, 38]. The experiment set in [47, 48] 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 [19] includes a point on a torus environment, an environment with a 5 link arm, and an environment with a planar manipulator with 2 arms holding an object (similar to our wheeled grasper). The experiments in [38] consist of a rigid body environment, a 7-dof arm robot, and a 12 dof closed chain environment which is similar to our wheeled grasper.

We include linkages and closed chains that are similar to the linkage and closed

*Reprinted with permission from "Sampling based motion planning with reachable volumes: Application to manipulators and closed chain systems." by Troy McMahon, Shawna Thomas, and Nancy M. Amato, 2014. Proc. IEEE Int. Conf. Intel. Rob. Syst. (IROS), pages 3705-3712, Copyright 2014 by IEEE.

chain problems in these papers (the only difference being that we use spherical joints while they use 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 while they only study problems with 20 dofs). Our experiment set also includes linkages and closed chains in environments with narrow passages (the tunnel and walls environments) while these other papers run there experiments either in totally free environments or 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 [39]. All computation was 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.

8.1.2 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 8.1.

Walls: The walls environment (Figure 8.1(a)) is a commonly used benchmark. It is a 19x4x4 unit^3 environment consisting of 3 chambers separated by 2 walls. Both walls have 1x1 openings allowing the robot to travel between the chambers. We run experiments with free flying chain linkages of varying dof (22–262) and single loop closed chains of (22dof–72dof). All chains have a total length of 2 units long and consist of links 0.02x0.02 units in width.

Environment	Robot Type	Dof
Walls	Chain	22,70,134,262,518,1034
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

Table 8.1: Combinations of environments and robots used in our experiments.

Tunnel: The tunnel environment (Figure 8.1(b)) is another commonly used benchmark. It is 39x10x1 unit^3 and consists of 2 chambers connected by a long narrow tunnel 1x1 units wide. Free flying chain linkages of varying dof (22–262) are all 1 unit long and consist of links 0.01x0.01 units in width. Single loop closed chains of varying numbers of links (9–37) have a total length of 1 unit.

Grid: Grid (Figure 8.1(c)) is a 14x14x14 unit^3 environment containing a set of 1x1x1 cube obstacles arranged in a grid separated by 2 units of free space. Two types of robots are investigated: a 16 joint (32 dof) fixed-based chain and two fixed-based tree-like robots comprising of an arm and a grasper formed by 2 subchains where one has 8 links in the arm and 4 links in each grasper yielding 32 dof and the other has 16 links in the arm and 8 links in each grasper yielding 64 dof.

WAM clutter: The WAM clutter(W-CL) environment(Figure 8.1(d)) consists of 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 (total 15 dofs). 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 dof (22–262) in a free environment.

Rods: The rods environment (Figure 8.1(e)) consists of 4 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 chain.

Loop-tree robot: The loop-tree (Lp-tr) robot (Figure 8.1(f)) consists of an 8-link central loop with 4 8-link branches attached to it. At the end of each branch another 8-link loop is attached giving this robot a total of 5 loops and 160 dof. Experiments are run in a completely free environment.

Wheeled grasper: We study a wheeled robot with 2 graspers attached to it (Figure 8.1(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 8.1(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). This is a scenario you would encounter in an industrial setting where a robot is operating a tool that uses an external power supply. We use two variations of this environment, one in which the robot consisted of 9 links (16 dof) with the cord attached to the 6th joint (cord-16) and another in which the robot consisted of 32 links (64 dof) with the cord attached to the 21st joint (cord-64). This environment demonstrates that our method can handle constraints on internal joints.

Fixed base grasper: The fixed base grasper environment (Figure 8.1(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 64-dof variations 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 8.1(j)) consists of a 22-dof closed chain (cc-22) or a 70-dof closed chain (cc-70). Constraints are applied to 3 of the chain's joints so that these joints must always be withing a small distance of each other.

Wheeled grasper with bucket: The grasper with bucket (Figure 8.1(k)) is a variation of the wheeled grasper environment in which the grasper is carrying a bucked 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 8.1(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 8.1(m)) consists of a Barrett WAM robotic arm which consists of a 6-dof arm with three graspers attached to it (total 15 dofs). 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 in order to grasp the object.

8.2 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 thy proportion of samples which 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.

We compare reachable volume sampling (Section 5.1) to uniform sampling [21] and an incremental sampling method, I-CD, which incrementally tests links along the chain starting at the base for collision before sampling the next link. I-CD detects invalid links as soon as they are sampled eliminating the need to sample the rest of the chain when collisions are found (see Algorithm 16). For closed chains we also compare to a CCD [43] sampler 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.

Algorithm 16 Incremental CD sampling method

```
Randomly sample translational/rotational coordinates
Test base for collision
For each joint, $j$
  Sample joint angles for $j$
  For each child link of $j$
    Test link for collision with links that have already been sampled
    Test link for collision with obstacles in environment
```

Uniform 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. 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.

8.2.1 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 8.2 compares the performance of each reachable volume, uniform and incremental CD check sampling in generating 2000 samples for various environments with chains and tree-like robots. Stars indicate methods that were unable to generate 2000 samples in the allotted 20 hours (e.g., uniform sampling for the tunnel environment with more than 70 dof or for the walls environment with more than 134 dof).

The sampler success rate (Figure 8.2(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 the distribution of reachable volume samples results in more collisions with obstacles (i.e. external collisions). However, as the dof of the problem increases, reachable volume sampling outperforms the other methods and in some cases is the only method able to generate roadmaps in the allotted time. Interestingly, the success rate of reachable volume sampling does not significantly decrease with problem dimension.

Figure 8.2(b) provides the time required for each method to generate 2000 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 less compact 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).

8.2.2 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 volumes is able to generate samples in many environments where other methods fail.

Figure 8.3 gives the performance of reachable volume sampling for robots containing closed chains. Again, 2000 valid, constraint-satisfying samples are created for each problem. Neither uniform sampling or 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 8.3(a)), yet reachable volume sampling is still able to generate valid, constraint-satisfying samples for single loops up to 262 dof and complex robots like the loop-tree with 160 dof. This trend is echoed in the increasing time required to generate such samples (Figure 8.3(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 and 134/262 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 sampling can be applied to problems such as the loop-tree robot where CCD sampling is not applicable.

8.2.3 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 8.4 gives the performance of reachable volume sampling for constrained systems. Again, 2000 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 8.4(a)). They also show that reachable volume sampling is able to produce samples in difficult environments where other methods fail.

Our timing results (Figure 8.4(b)) show the running time of reachable volume sampling is less than uniform sampling with filtering and significantly less than CCD. This is because uniform sampling with filtering generates many samples which must be discarded because they do not satisfy constraints, and CCD requires significant time to step samples towards constraints. In comparison, reachable volumes 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, however the benefits in terms of connectivity and local planner success still make reachable volume sampling preferable in this environment.

In the grid and WAM environments both Uniform 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 consists of tree-like graspers with constraints applied to their end effectors. While a method like CCD can converge to an end effector constraint, neither CCD or Uniform with filtering can endure that the base of the fingers of the grasper is in a position where the other graspers can reach their associated constraints. In these problems you need a more powerful method like reachable volumes which can position the base of the graspers in a position where all of the end effectors can reach their constraints. The CCD method also fails in the higher dof c-64 and cc22/70 environments.

8.2.4 Coverage and Sample Distribution

We next study the distribution of reachable volume samples in order 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 [43].

Reachable volume sampling samples positional and rotational coordinates in the same manner as uniform, 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 4 different environments; a 4 link open chain, a 4 link open chain with links of unequal length, a 5 link closed chain and a 4 link chain with an end effector constraint (see Figure 8.5). 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 6.2.1). In the open chain environments we compared to uniform sampling while in the closed chain and constrained environment we compared to CCD.

To evaluate coverage and sample distribution we generate 100 samples using each method. For each sample we then computed and plotted the Euclidean distance between each pair of non-adjacent joints (Figures 8.6, 8.7, 8.8 and 8.9). Studying the range of distances and comparing them to the feasible range of distances for each robot 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 joints. Studying the distribution of these ranges will indicate if the samples are evenly distributed. Ideally the distances between each pair of joints should be distributed uniformly over the range of feasible distances.

Figure 8.6 shows a scatter plot of the distances separating all pairs of non-adjacent joints in a 4 link open chain consisting of .25 unit links connected by spherical joints (see Figure 8.5(a)). We first observe that reachable volumes with an end effector 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 0 and 4 (denoted as (0,4) in Figure 8.6, where reachable volume sampling with end effector first gives better coverage over the larger distance ranges (.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. These results indicate that an end effector first ordering produces better coverage over the regions of sample space which 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 range.

Reachable volume sampling with root first also produces a similar distribution to uniform with the exception of (0,4). For (0,4) it gets similar or better coverage over the middle of the distance range (.2 to .8) but worse coverage outside of this range.

Figure 8.7 is a plot of the distances for a 4 link open chain (Figure 8.5(b)) in which the first, second and fourth links are .25 units long, and the third link is 1 unit long. Overall both reachable volumes methods give better coverage than uniform. Reachable volumes with end effector first 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 volumes with root first 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 8.8 is a plot of distances for a single loop closed chain comprised of five .25 unit links connected by spherical joints (Figure 8.5(c)). We observe that both reachable volumes methods give better coverage of the lower distance ranges than CCD. This is particularly noticeable for the (J_0,J_3) and (J_1,J_4) ranges with end effector first and the (J_0,J_2) and (J_1,J_4) ranges with root first. These results show that reachable volume sampling gives better coverage over the regions of sample space that correspond to highly deformed samples (where nonadjacent joints of the closed chain are close together).

We also observed the 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 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 epsilon approaches 0, making it unfeasible to generate samples that are more exact then those shown (with an ϵ value of .015). In contrast, reachable volume sampling can efficiently generate samples that exactly satisfy a problem's constraints (as shown by the analysis of the running times of reachable volume and CCD sampling presented in Section 8.2).

Figure 8.9 is a plot of distances for a chain of four .25 unit links whose end effector is constrained to a point that is .75 from the root of the chain (Figure 8.5(d)). We observe that both reachable volumes methods produced distributions that were similar to that of CCD. The only exception was the (J_0,J_2) distance of root first 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.

8.3 Reachable Volume Local Planner and Distance Metric in Practice[†]

In this section we evaluate the reachable volume local planner and 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 to those produced by straight line, and the connections produced using the reachable volume distance metric to those produced using scaled Euclidean.

[†] Reprinted with permission from "Reachable volume RRT." by Troy McMahon, Shawna Thomas, and Nancy M. Amato, 2015. Proc. IEEE Int. Conf. Robot. Autom. (ICRA), pages 2977-2984, Copyright 2015 by IEEE.

8.3.1 Performance

The primary advantage of the reachable volume local planner is that it generates paths that satisfy the problem's constraints while other methods such as straight line do not. We compare the reachable volume local planner and distance metric to 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 2000 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 8.10 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.

8.3.2 Connectivity

In this section we study the connections produced by the reachable volume local planner and distance metric. 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 8.1(a)) which we ran in

	Straight Line	RVLP
Scaled Euclidean	w-22, t-22	w-22, t-22, w-cc
RVDM	N/A	w-22, w-cc

Table 8.2: Combinations of local planner and distance metric run for each environment

combination with a 22 dof chain (w-22) and a 22 dof closed chain (w-cc), and the tunnel environment (Figure 8.1(b)) which we ran in combination with a 22 dof chain (t-22). In each of these problems we generated 2000 reachable volume samples. We then connected these nodes to their 8 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 8.2. Note that we did not run experiments using the reachable volume local planner 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. Also note that we did not run experiments using straight line for closed chains because straight line can't generate paths that maintain closure constraints, making it unsuited 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 8.11 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 reachable volume distance metric (RVDM) to those generated using scaled Euclidean (se). Our results (Figure 8.12) show there is a significant difference in the edges which indicates that neighbors being selected by 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 reachable volume distance metric and scaled Euclidean.

8.4 Evaluation of Reachable Volumes for PRM Construction

In this section we study how reachable volumes can be applied to roadmap construction. We show that reachable volumes produce better connected roadmaps than existing methods and that they are capable of solving many difficult high dof problems which 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 [9] and local planning is done using a binary straight line local planner [21].

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 (CC), and the number of connected components in the roadmap.

Figure 8.13(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 (CC) 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 CC are more desirable. Figure 8.13(b) displays that roadmaps using reachable volume sampling produce a greater percentage of samples in the largest CC than the other methods. This suggests that reachable volume sampling is doing a better job of finding connections between various areas of C-free, 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 size of the largest CC, the *number of connected components* (Figures 8.13(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 or I-CD sampling. As with the local planner success, this indicates that reachable volume samples are easier to connect than samples produced by the other methods.

8.4.1 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 and largest cc of roadmaps generated using the reachable volume sampler across a variety of n values. 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.

In the 22-dof walls environment (Figure 8.14(a,b)), the local planner success of both methods is significant, however the local planner success with reachable volume samples is consistently higher than with uniform sampling. The largest CC size for the 2 sampling methods is similar across the range of n values we studied, and generally increases with n, indicating that adding more nodes is improving 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 cc size that is larger than .4 implies that a method has connected 2 of the chambers and a largest cc size that is greater that .7 implies that a method has connected all three chambers. Both methods produce roadmaps where the largest cc size is significantly larger than .4, which indicates that both methods are producing connections between chambers in the environment.

In the 70-dof walls environment (Figure 8.14(c,d)), the Reachable volume sampler consistently produced roadmaps with a considerably higher local planner success rate then uniform sampling, even for large n values. It also consistently produced roadmaps that are more connected than uniform sampling, resulting in larger largest CCs. The largest cc size of the reachable volumes roadmaps increased significantly over the range of n values that we studied. For the larger values in this range, the largest cc size is significantly larger than .4, which indicates that it is consistently making connections between the chambers of the environment (as shown in Figure 8.17). In comparison, the largest cc of the uniform roadmaps remained at .3 over the entire range of n values we studied

In the 70-dof tunnel environment (Figure 8.15), the reachable volumes sampler also results in a higher local planner success than uniform sampling across the entire range of n values we studied. The largest CC size of the reachable volumes roadmap is significantly larger than uniform sampling over the entire range of n values 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 spaces is located in the tunnel region. The largest CC size of the reachable volumes roadmaps is significantly larger that .25, which indicates that it is making connections into the tunnel (as demonstrated in Figure 8.18). The largest CC 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 1034-dof chains, and the tunnel environment with the 262-dof chain (Figures 8.16) demonstrate 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 that is comparable to the lower dof problems. The largest CC 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.

8.5 Evaluation of Reachable Volumes for RRT Construction[‡]

[‡] Reprinted with permission from "Reachable volume RRT." by Troy McMahon, Shawna Thomas, and Nancy M. Amato, 2015. Proc. IEEE Int. Conf. Robot. Autom. (ICRA), pages 2977-2984, Copyright 2015 by IEEE.

In this section we study how reachable volumes can be applied to RRT construction. We first evaluate the range of parameter settings to determine which settings produce good results. We then demonstrate that RVRRTs are capable of solving problems more efficiently then existing methods and of solving problems existing methods cannot.

8.5.1 RV-Expand Parameter Study

We first evaluate the range of parameter settings for RV-Expand to determine which produce good results. RV-Expand takes the following parameters:

- **Order of Reachable Volume Computation:** The possible orderings are **End Effector First**, **Root First**, and **Binary**. The effects of the ordering are discussed in Section 5.2.

- **Repositioning Policy:** This policy determines how joints that are no longer in their reachable volumes are repositioned (line 5 of Algorithm 12). We consider two policies: select a **Random** point in the new reachable volume and select the point in the new reachable volume that is **Closest** to the original position of the joint.

- **Joint Selection Policy:** This policy determines how to select the perturbed joint (line 4 of Algorithm 15). We study two policies: select a **Random** joint and select the joint that is **Most Distant** from its q_{ran} counterpart.

- δ**:** The step size used when generating q_{new}. To facilitate comparison across different environments, δ is normalized by the environment diameter.

- **Scaling Factors:** s indicates the relative weighting of reachable volume distance and rigid body distance while s_{rot} indicates the relative weighting of the translational and rotational coordinates (see Section 5.4).

We first evaluate the different combinations of reachable volume computation order, joint selection policies, and repositioning policies, see Table 8.3. We run each combination across a variety of δ, s and s_{rot} values and select the settings which solve a problem with the fewest number of nodes. We will study these parameters in detail next.

Method	Computation Order	Repositioning	Joint Selection
RVRRT-1	EndEffectorFirst	Closest	Closest
RVRRT-2	EndEffectorFirst	Closest	Random
RVRRT-3	EndEffectorFirst	Closest	MostDistant
RVRRT-4	EndEffectorFirst	Random	Closest
RVRRT-5	EndEffectorFirst	Random	Random
RVRRT-6	EndEffectorFirst	Random	MostDistant
RVRRT-7	RootFirst	Closest	Closest
RVRRT-8	RootFirst	Closest	Random
RVRRT-9	RootFirst	Closest	MostDistant
RVRRT-10	RootFirst	Random	Closest
RVRRT-11	RootFirst	Random	Random
RVRRT-12	RootFirst	Random	MostDistant
RVRRT-13	Binary	Closest	Closest
RVRRT-14	Binary	Closest	Random
RVRRT-15	Binary	Closest	MostDistant
RVRRT-16	Binary	Random	Closest
RVRRT-17	Binary	Random	Random
RVRRT-18	Binary	Random	MostDistant

Table 8.3: RVRRT variations from different policy combinations.

Figure 8.19 shows the number of nodes and running time required for each combination to solve l-tunnel (l-tun), 70-dof walls (w-70), and rods. Overall, methods with root-first or binary reachable volume computation orders outperformed methods with end effector first. Methods with random joint selection tended to do better than methods with closest or most distant joint selection. Methods with random repositioning also did better than closest repositioning.

We next observe that methods 11, 12, and 14 (RVRRT-11, RVRRT-12 and RVRRT-14) gave the best results. RVRRT-11 solves all environments, requires the least running time to solve rods, and performs well in other environments. RVRRT-14 also solves all environments, gives the best performance for l-tun, and was one of the most efficient methods in walls. RVRRT-12 gives the best performance in walls and the second best performance in l-tun, although it does not solve rods.

We also ran experiments using δ values ranging from .001 to 10, s values ranging from .025 to .075, and s_{rot} values ranging from .025 to .975. Table 8.4 shows the best δ, s, and s_{rot} values for the selected methods in each environment. The best δ values were similar when the methods were applied to the same environment but varied greatly across environments. The best s value was consistently around .9, and the best s_{rot} value was always between .075 and .25. In our remaining experiments we use a $s = .9$ and a $s_{rot} = .1$, and tune δ to the environment.

Method	Environment	δ	s	s_{rot}
RVRRT-11	l-tun	12.5	0.9	0.1
	walls-70	5	0.9	0.1
	rods	0.788	0.925	0.075
RVRRT-12	l-tun	20	0.9	0.2333
	walls-70	7	0.9	0.75
	rods	-	-	-
RVRRT-14	l-tun	15.875	0.9	0.2
	walls-70	6.4	0.9	0.75
	rods	0.6295	0.925	0.1625

Table 8.4: Best δ, s, and s_{rot} values for selected methods.

8.5.2 RVRRT in Practice[§]

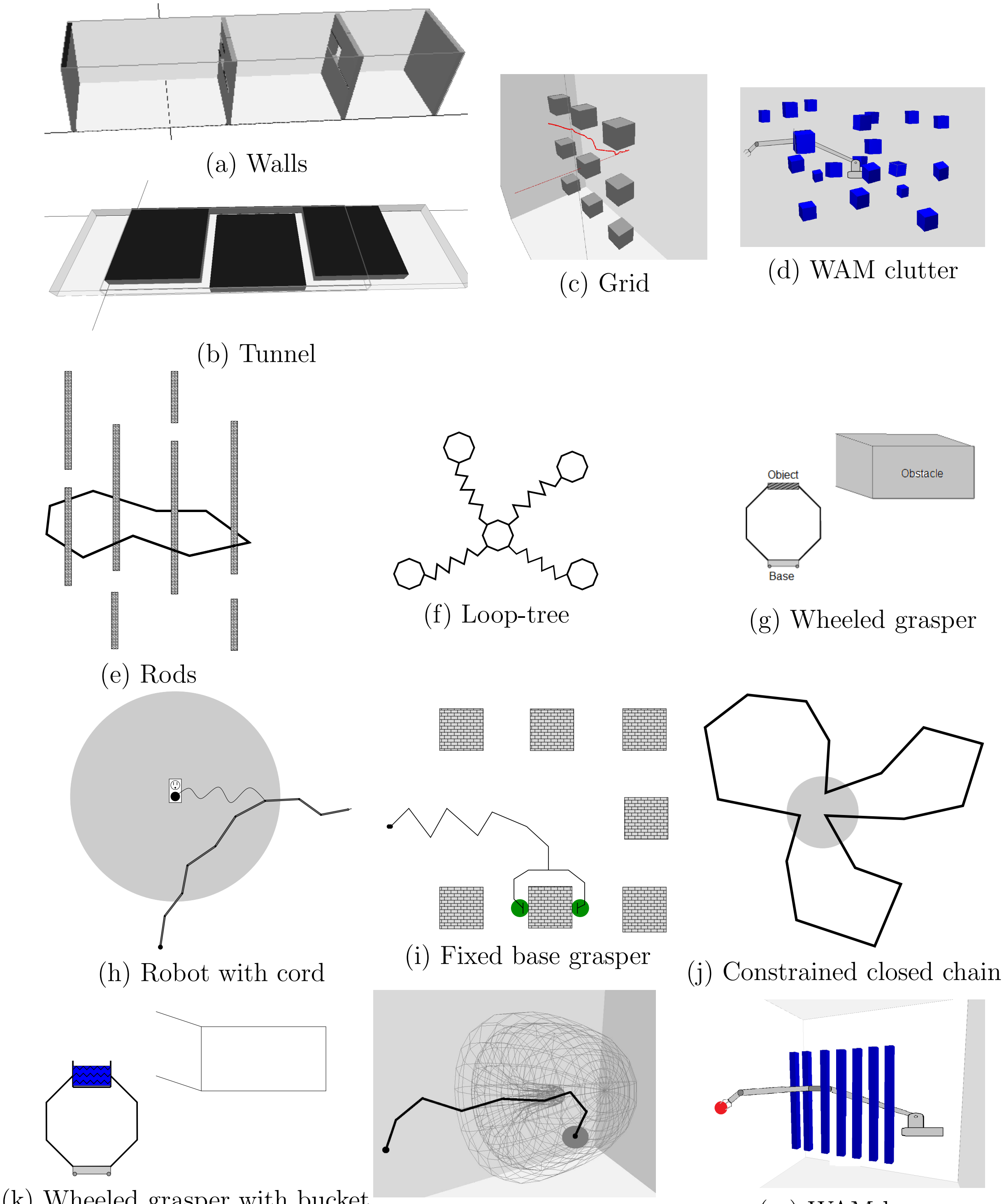
Here we compare the RVRRT variations selected in Section 8.5.1 to the RRT [24] and DDRRT [47, 48] methods. As in the previous section, we study the number of nodes and the running time required to solve a problem. Our results (Figures 8.20 and 8.21) demonstrate that RVRRTs are capable of solving problems more efficiently then existing methods and of solving problems existing methods cannot in both unconstrained and constrained systems.

We first observe that RVRRT variations are able to solve all of the problems that RRTs and DDRRTs could solve. Furthermore, RVRRT consistently required fewer nodes to solve these problems than RRT or DDRRT (Figures 8.20(a) and 8.21(a)). In some cases, such as st-22 and m-22, the RVRRT variations require substantially fewer nodes. This is important because roadmaps with fewer nodes require less memory and are cheaper to query. We next observe that the RVRRT variations are more efficient than RRT and DDRRT in that they require fewer collision detection calls (Figures 8.20(b) and 8.21(b)). The running time of RVRRT is generally higher than RRT in low dof problems but comparable to RRT and DDRRT in higher dof problems (Figures 8.20(c) and 8.21(c)). Finally, we observe that RVRRTs are able to solve many difficult problems, such as the l-tun, r-70, and r-cc, that RRTs and DDRRTs are not able to. RVRRTs are also the only methods that were able to solve the high dof w-134 and st-134 environments.

There are a number of explanations for why RVRRTs outperform RRTs and DDRRTs. One reason is that reachable volume stepping tends to move the joints of the robot in the same direction in workspace. This allows RVRRTs to expand more

[§] Reprinted with permission from "Reachable volume RRT." by Troy McMahon, Shawna Thomas, and Nancy M. Amato, 2015. Proc. IEEE Int. Conf. Robot. Autom. (ICRA), pages 2977-2984, Copyright 2015 by IEEE.

rapidly into more distant regions of the workspace. Another reason is that RVRRTs tend to produce nodes where the orientation of the joints is more uniform. This trend was observed in [29] and was shown to produce samples that were less likely to be invalid due to self-collision. A third reason is that a problem's constraints are incorporated into reachable volumes, so new nodes will always conform to them. A fourth reason comes from using reachable volume sampling, which generates q_{ran} samples over the constraint satisfying subset of C-space. The growth of the RVRRT will therefore be biased towards the unexplored regions of the constraint-satisfying subset of C-space.



(a) Walls

(b) Tunnel

(c) Grid

(d) WAM clutter

(e) Rods

(f) Loop-tree

(g) Wheeled grasper

(h) Robot with cord

(i) Fixed base grasper

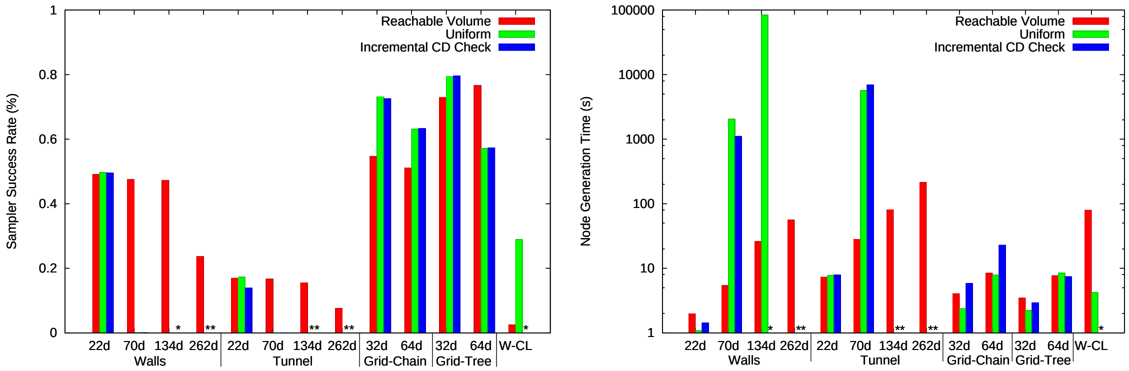
(j) Constrained closed chain

(k) Wheeled grasper with bucket

(l) Bug-trap cleaner

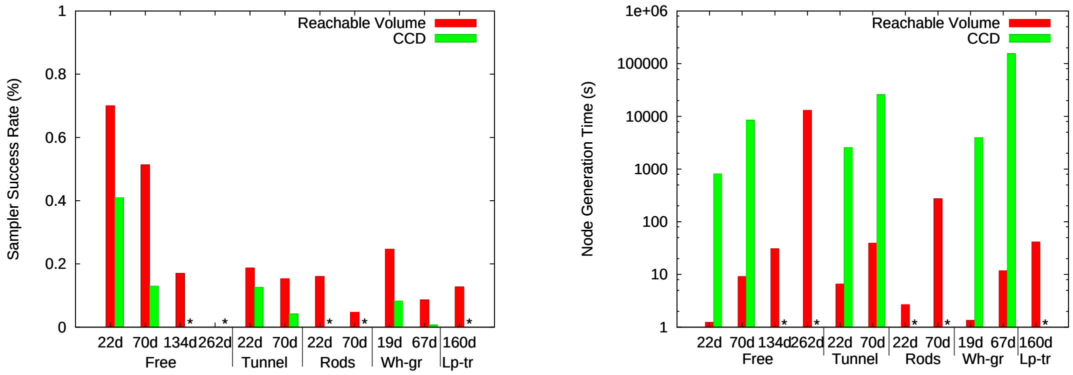
(m) WAM bars

Figure 8.1: Environments studied.



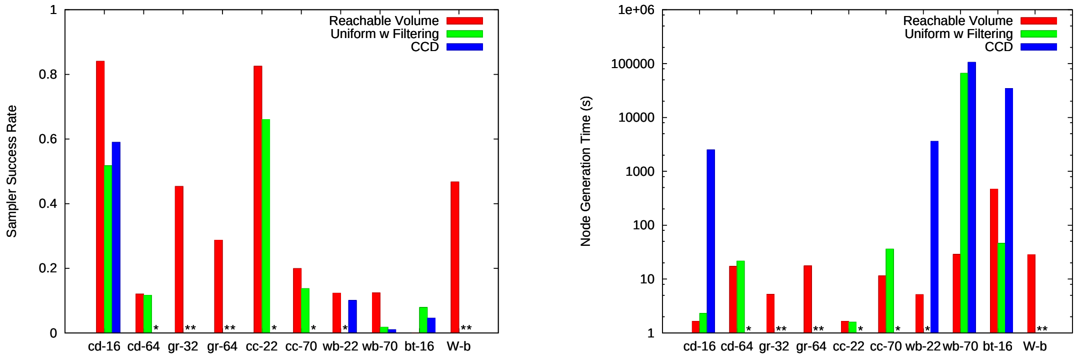
(a) Sampler Success Rate (b) Node Generation Time

Figure 8.2: Experimental results for chains and tree-like robots in various environments for 2000 samples. Stars indicate methods unable to generate samples in the allotted time. Note that (b) uses a log scale.



(a) Sampler Success Rate (b) Node Generation Time

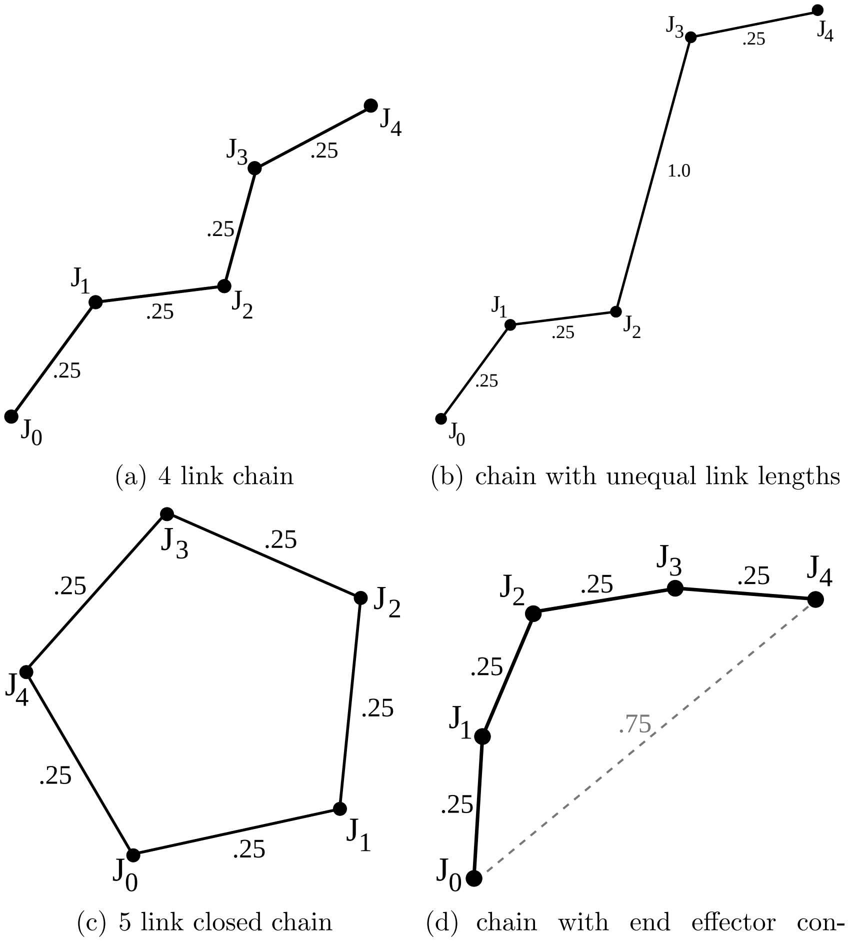
Figure 8.3: Reachable volume performance for closed chains in the following environments for 2000 samples: free, tunnel, rods, wheeled grasper (Wh-gr), and the loop-tree robot (Lp-tr). Uniform sampling and I-CD are infeasible for these robots. Note that (b) uses a log scale.



(a) Sampler Success Rate

(b) Node Generation Time

Figure 8.4: Experimental results for 2000 samples in various constrained systems. Stars indicate methods unable to generate samples in the allotted time or were not applicable. Note that (b) uses a log scale.



(a) 4 link chain

(b) chain with unequal link lengths

(c) 5 link closed chain

(d) chain with end effector constraint

Figure 8.5: Robots used in joint distance study.

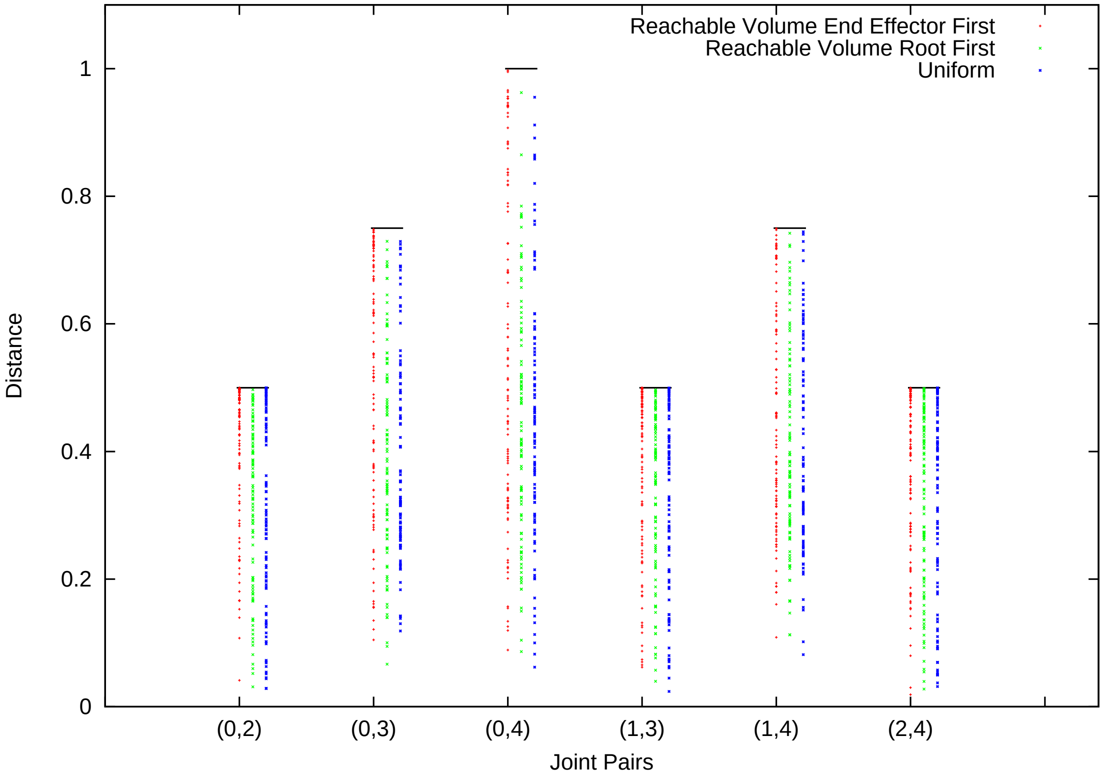


Figure 8.6: Distance between pairs of joints for 100 reachable volume/uniform samples of a 4 link open chain with links of length .25. Horizontal lines indicate the maximum distance between each joint pair.

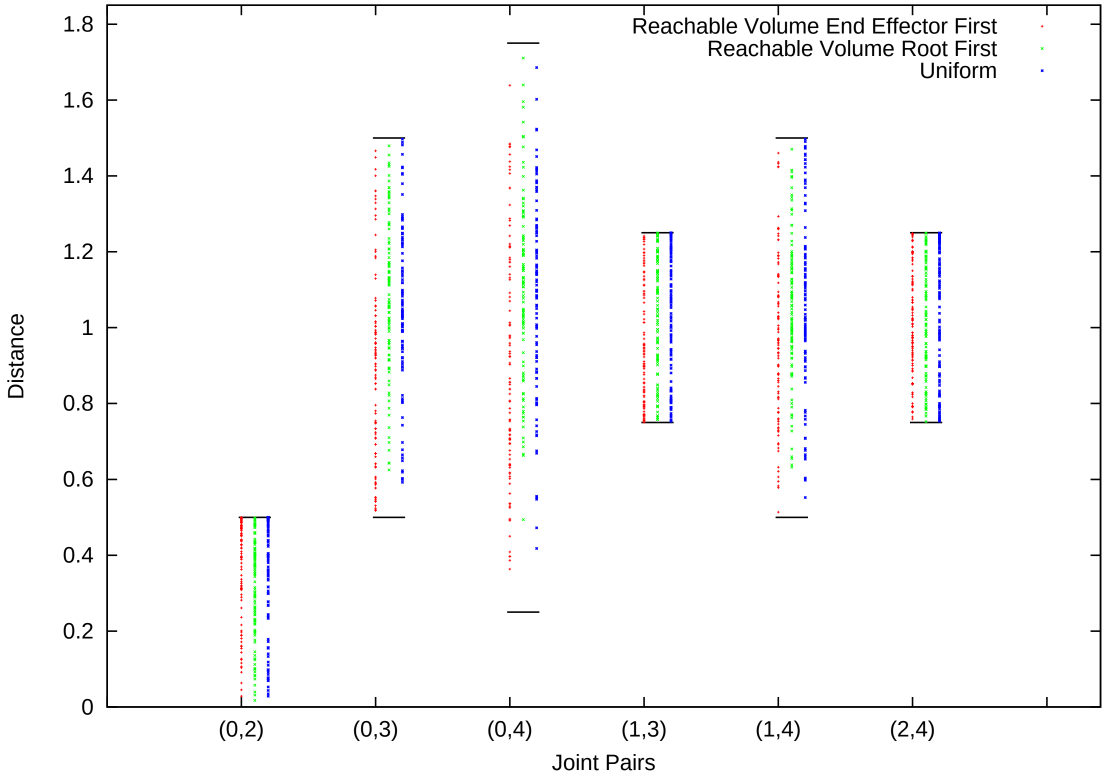


Figure 8.7: Distance between pairs of joints for 100 reachable volume/uniform samples of a 4 link open chain where the first, second and fourth links have a length of .25, and the third link has a length of 1. Horizontal lines indicate the maximum and minimum distance between each joint pair.

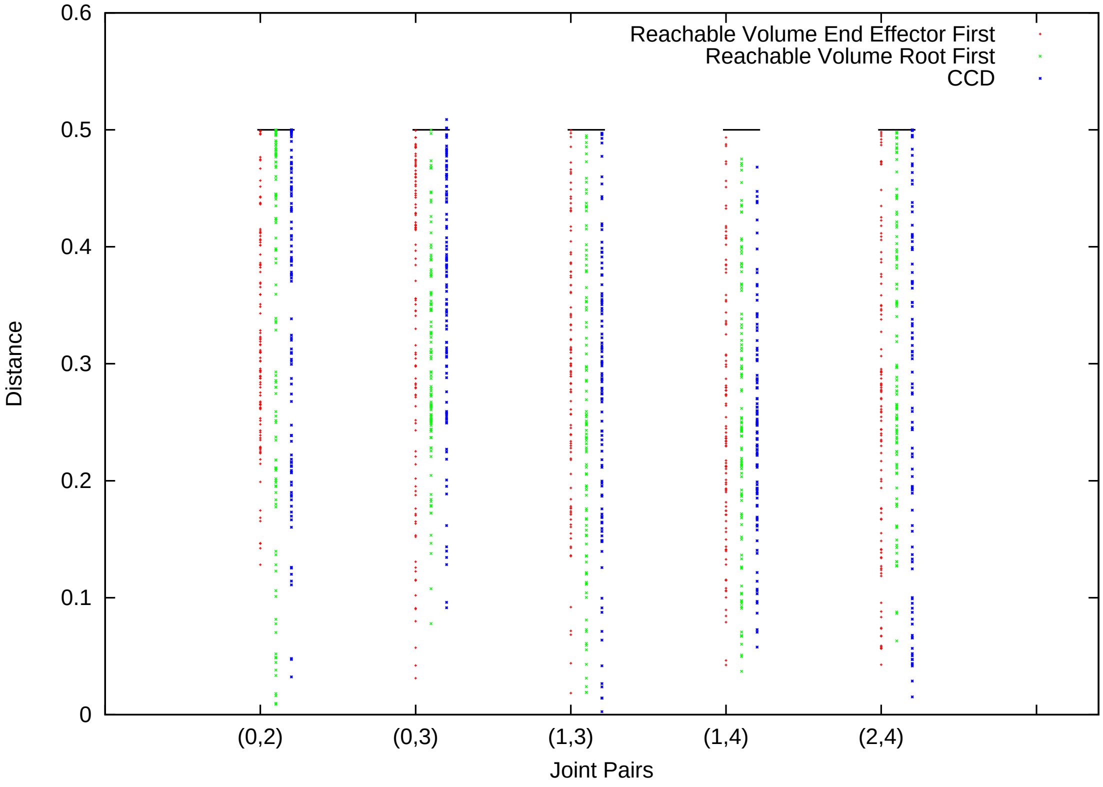


Figure 8.8: Distance between pairs of joints for 100 reachable volume/uniform samples of a 5 link closed chain with links of length .25. Horizontal lines indicate the maximum distance between each joint pair.

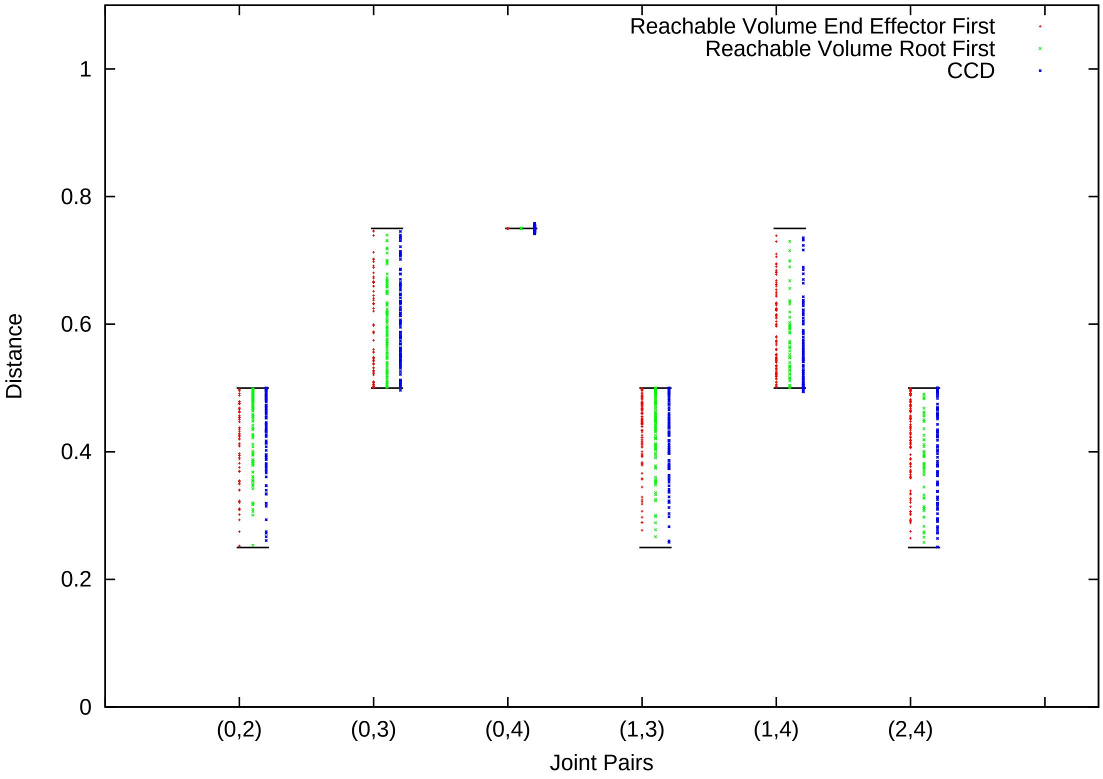
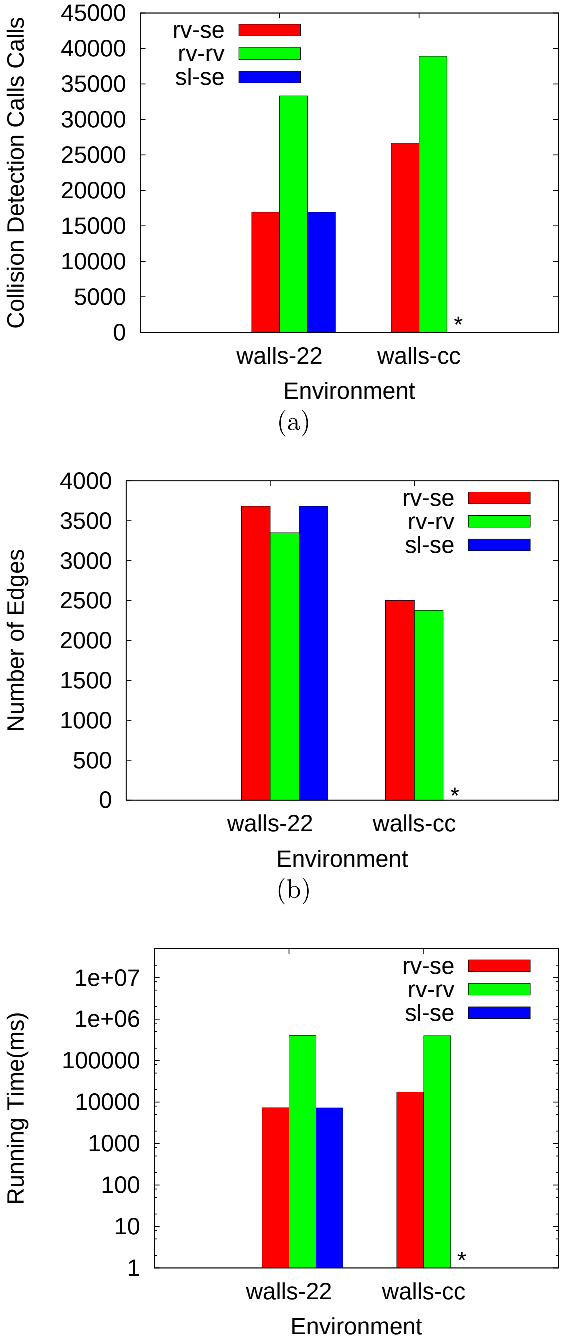


Figure 8.9: Distance between pairs of joints for 100 reachable volume/uniform samples of a 4, .25 unit link open chain with its end effector constrained to be .75 units from its base. Horizontal lines indicate the maximum and maximum distance between each joint pair.



(a)

(b)

(c)

Figure 8.10: (a) Collision detection calls, (b) number of edges, and (c) connection time for roadmaps constructed 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) when applicable (* denotes when sl-se is not applicable).

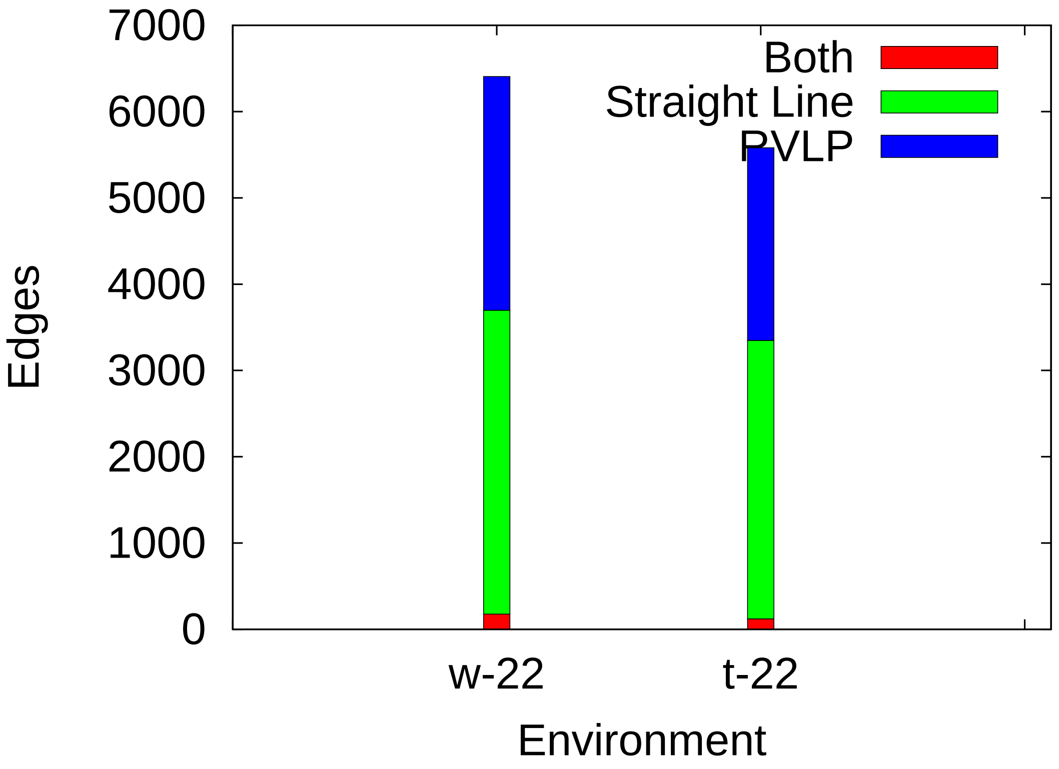


Figure 8.11: Edge difference between reachable volumes local planning (RVLP) and straight line (sl) using scaled Euclidean distance.

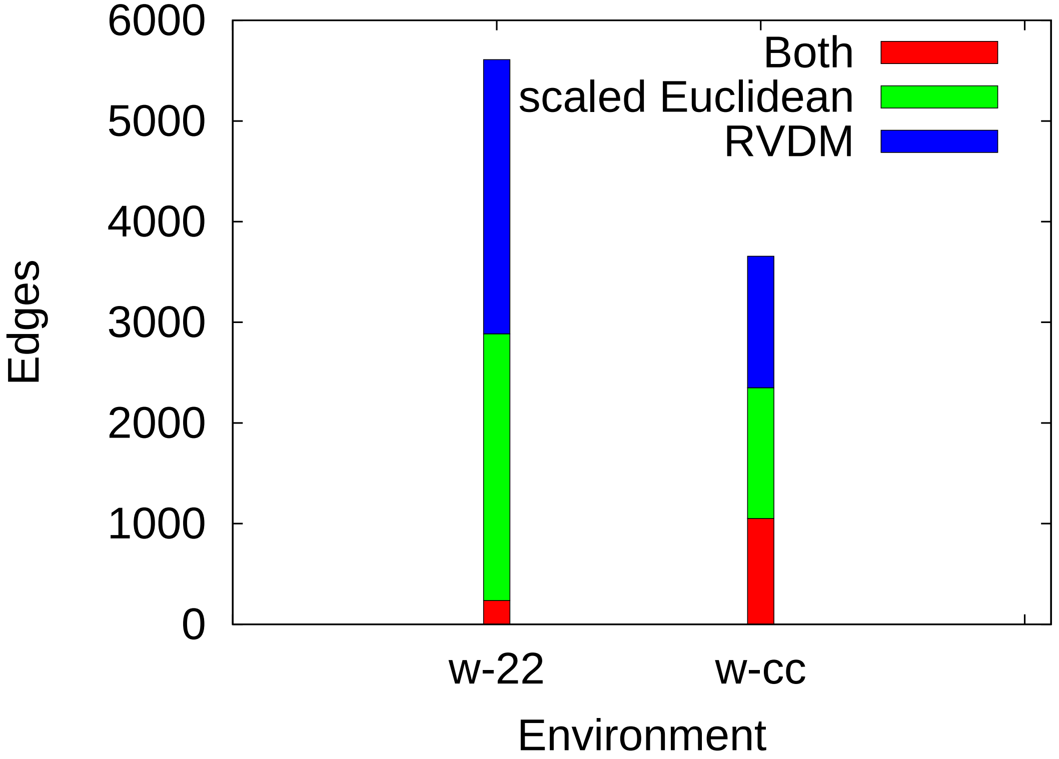
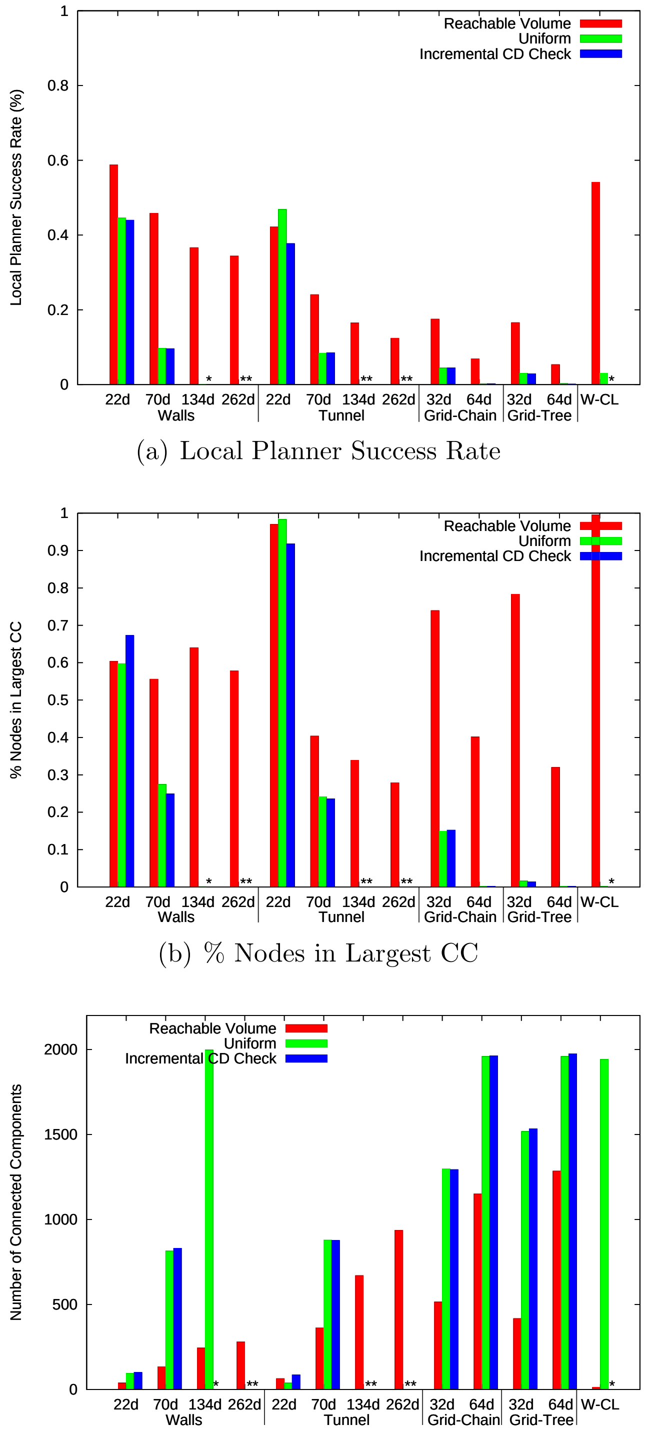


Figure 8.12: Edge difference between reachable volume distance (RVDM) and scaled Euclidean (se) using the reachable volume local planner.

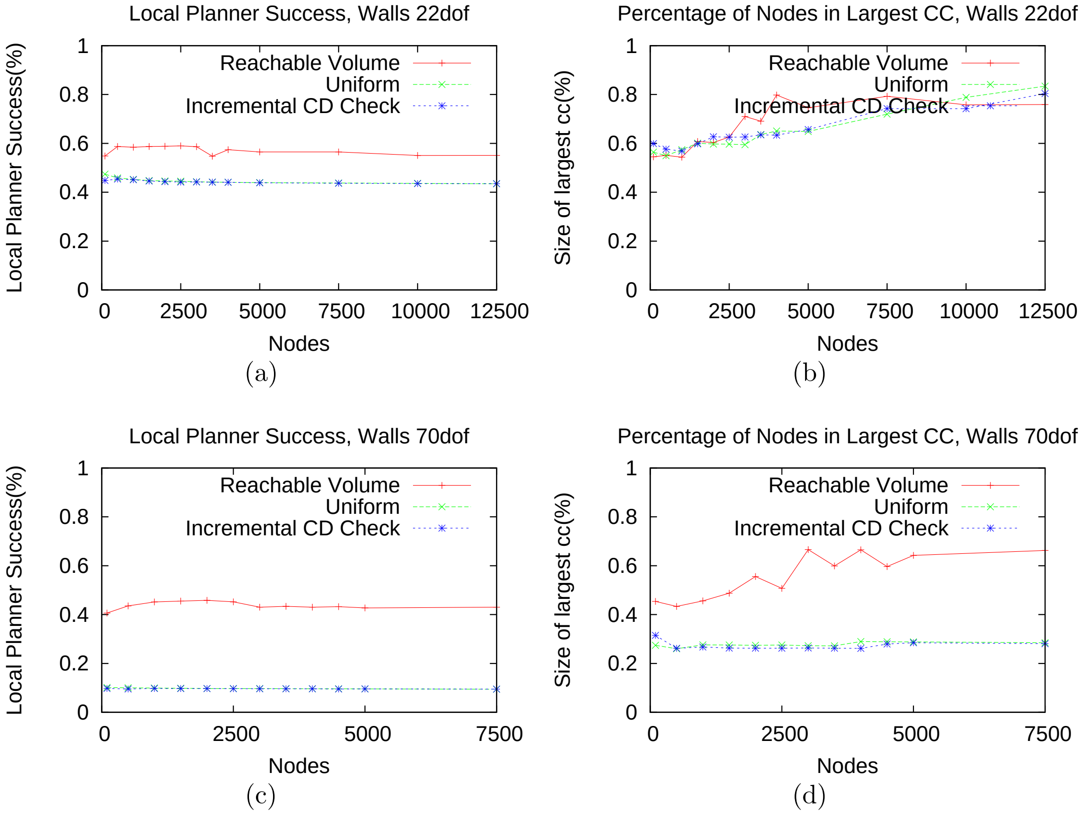


(a) Local Planner Success Rate

(b) % Nodes in Largest CC

(c) Number of CCs

Figure 8.13: Experimental results for chains and tree-like robots in various environments for 2000 samples. Stars indicate methods unable to generate samples in the allotted time.



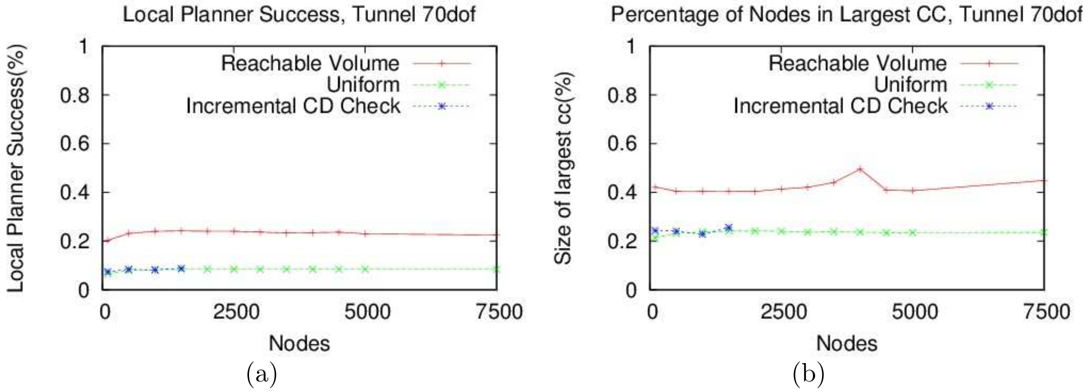
(a)

(b)

(c)

(d)

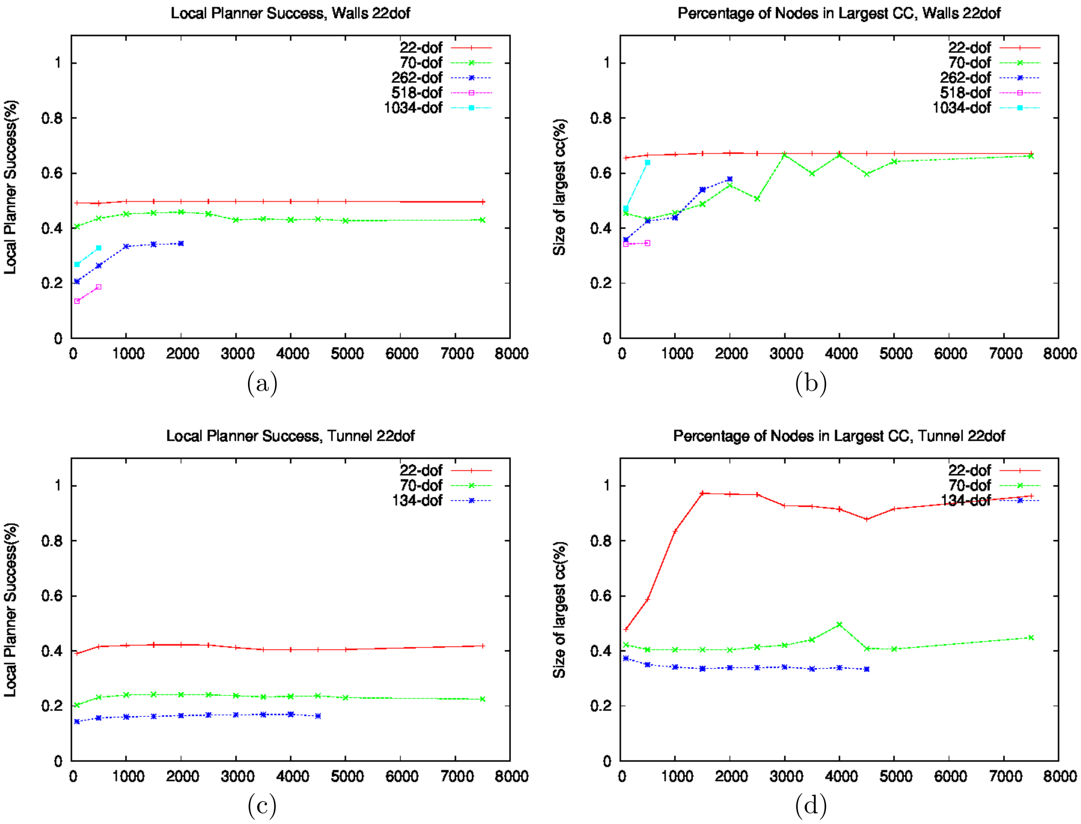
Figure 8.14: Local planner success and size of largest connected component for 22 and 70 dof chains in the walls environment.



(a)

(b)

Figure 8.15: Local planner success and size of largest connected component for 70 dof chain in the tunnel environment.



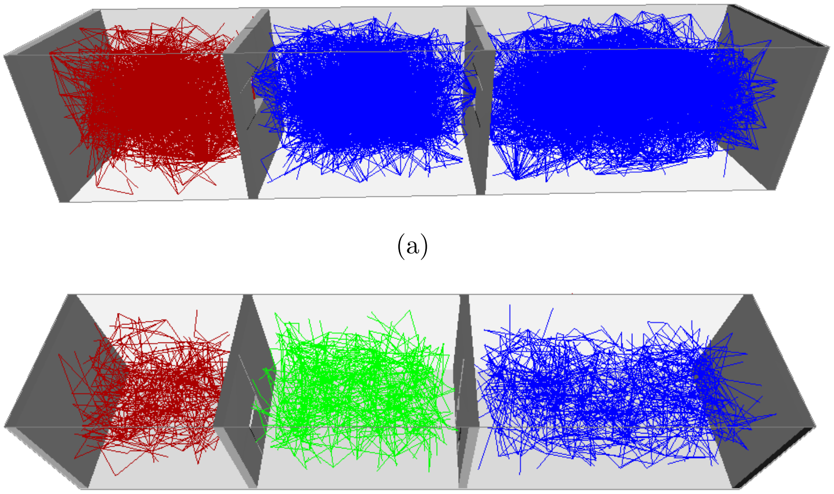
(a)

(b)

(c)

(d)

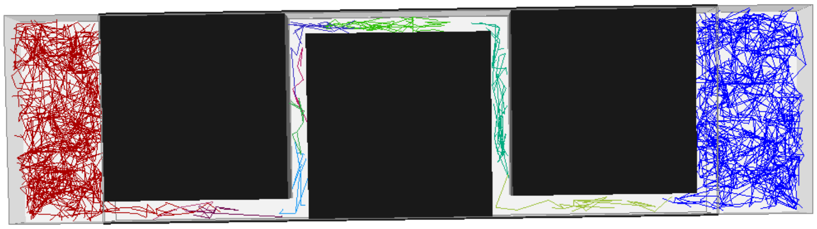
Figure 8.16: Evaluation of how local planner success and size of the largest CC scales with roadmap size in walls (a,b) and tunnel (c,d) environments with robots ranging from 22-dof to 1034-dof.



(a)

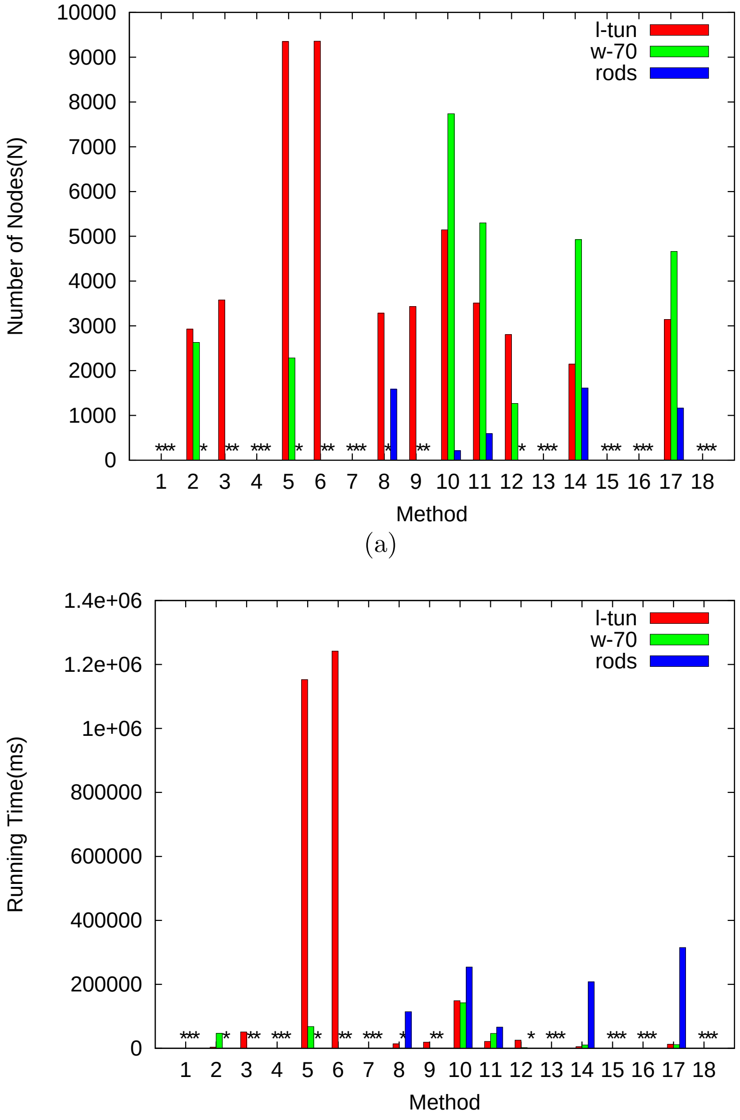
(b)

Figure 8.17: Sample images of the connected components for the 70-dof chain in the walls environment with 3500 samples. Notice that the reachable volume sampler(top) connects 2 of the chambers while the uniform sampler(bottom) does not.



(a)

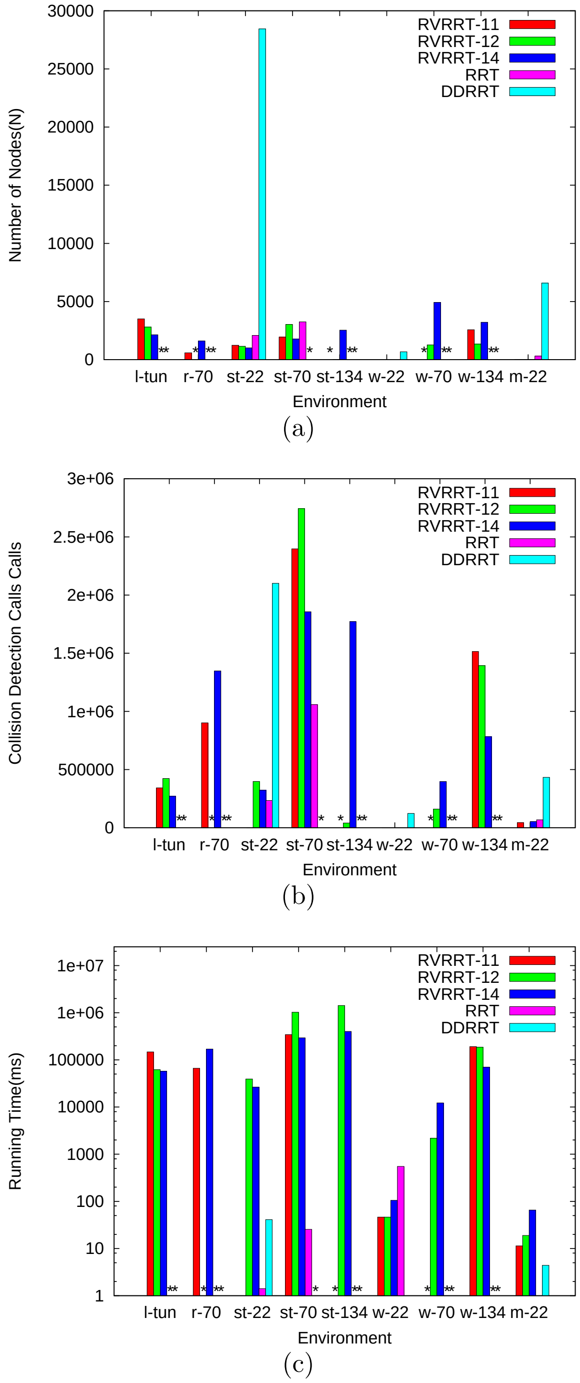
Figure 8.18: Sample images of the connected components for the 70-dof chain in the tunnel environment with 3500 reachable volume samples. Notice that this roadmap includes connected components in the tunnel region of the environment.



(a)

(b)

Figure 8.19: (a) Number of nodes and (b) running time required for RVRRT variants (see Table 8.3) in the l-tun, walls, and rods environments. *s indicates that a method was unable to find a solution.

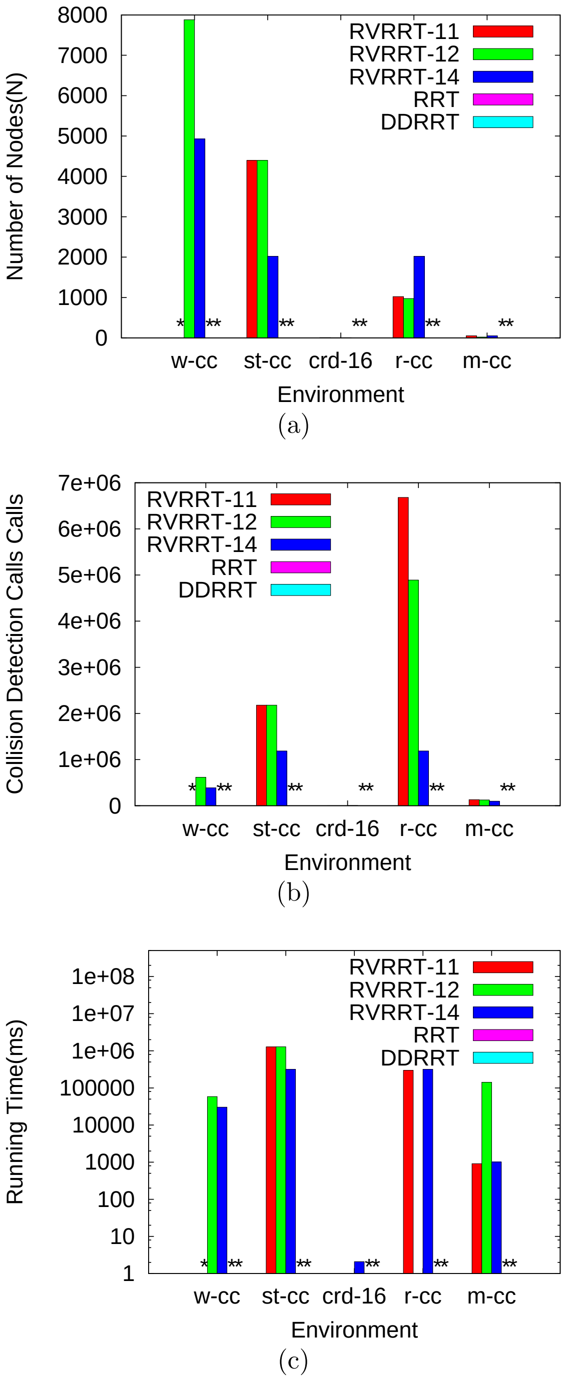


(a)

(b)

(c)

Figure 8.20: (a) Number of nodes, (b) collision detection calls and (c) running time required for RRT, DDRRT and 3 RVRRT variations in environments without constraints. *s indicates that a method was not able to find a solution.



(a)

(b)

(c)

Figure 8.21: (a) Number of nodes, (b) collision detection calls and (c) running time required for RRT, DDRRT and 3 RVRRT variations in environments with constraints. *s indicates that a method was not able to find a solution or could not be applied to the problem.

9. CONCLUSION

This dissertation introduces a new concept, reachable volume, that is 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 some cases trivial. We show that reachable volumes have a $O(1)$ complexity and can be computed in linear time in problems without constraints. In constrained problems, we show that the complexity is dependent on the complexity of the problem's constraints.

We introduce tools and techniques to extend the state of the art sampling based motion planning algorithms to RV-space. We present 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 show 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 also show that PRMs constructed using reachable volumes are probabilistically complete.

We demonstrate that reachable volume sampling can be applied to a wide variety of problems including high degree of freedom chains, tree-like linkages, closed chains, and combinations of them. We show results for constrained problems with as many as 70 degrees of freedom and for unconstrained problems with as many as 1034 degrees of freedom. We show 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. In contrast most previous methods either cannot be applied to these problems, do not produce quality solutions or have a significantly higher running time. We also show that it is applicable to a wide variety of constrained systems including problems which existing methods cannot solve. Our next step will be to explore applying reachable volumes to other problems. We are particularly interested in applying it to computational biology problems such as protein folding. These problems have a large number of degrees of freedom and are likely well suited for reachable volume sampling. We also plan to apply reachable volume sample to folding robots.

We also plan to explore how reachable volumes can improve control and interaction with high degree of freedom robots. We plan to further develop the reachable volume visualization tool so that it can be used to provide feedback that will assist in robot control. We also plan to apply reachable volumes and reachable volume sampling to user guided motion planning where reachable volumes can be used to guide planning and to provide feedback.

REFERENCES

[1] N. M. Amato, O. B. Bayazit, L. K. Dale, C. V. Jones, and D. Vallejo. Choosing good distance metrics and local planners for probabilistic roadmap methods. *IEEE Trans. Robot. Automat.*, 16(4):442–447, August 2000.

[2] Peter Anderson-Sprecher and Reid Simmons. Voxel-based motion bounding and workspace estimation for robotic manipulators. In *International Conference on Robotics and Automation*, pages 2141–2146, 2012.

[3] J. Cortés and T. Siméon. Probabilistic motion planning for parallel mechanisms. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 4354–4359, Taipei, Taiwan, 2003.

[4] J. Cortés, T. Siméon, and J. P. Laumond. A random loop generator for planning the motions of closed kinematic chains using PRM methods. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 2141–2146, Washington, DC, 2002.

[5] Juan Cortés and Thierry Siméon. Sampling-based motion planning under kinematic loop-closure constraints. In *Algorithmic Foundations of Robotics VI*, pages 75–90. Springer, Berlin/Heidelberg, 2005. book contains the proceedings of the International Workshop on the Algorithmic Foundations of Robotics (WAFR), Utrecht/Zeist, The Netherlands, 2004.

[6] John J. Craig. *Introduction to Robotics: Mechanics and Control*. Addison-Wesley Publishing Company, Reading, MA, 2nd edition, 1989.

[7] Willow Garage. PR2 Robot, 2016. www.willowgarage.com.

[8] Maxim Garber and Ming C. Lin. Constraint-based motion planning using Voronoi diagrams. In *Algorithmic Foundations of Robotics V*, pages 541–558.

Springer, Berlin/Heidelberg, 2003. book contains the proceedings of the International Workshop on the Algorithmic Foundations of Robotics (WAFR), Nice, France, 2002.

[9] S. Gottschalk, M. C. Lin, and D. Manocha. OBB-tree: A hierarchical structure for rapid interference detection. *Comput. Graph.*, 30:171–180, 1996. Proc. SIGGRAPH '96.

[10] L. Han and N. M. Amato. A kinematics-based probabilistic roadmap method for closed chain systems. In *New Directions in Algorithmic and Computational Robotics*, pages 233–246. A. K. Peters, Boston, MA, 2001. book contains the proceedings of the International Workshop on the Algorithmic Foundations of Robotics (WAFR), Hanover, NH, 2000.

[11] L. Han, L. Rudolph, J. Blumenthal, and I. Valodzin. Stratified deformation space and path planning for a planar closed chain with revolute joints. In *Proc. Int. Workshop on Algorithmic Foundations of Robotics (WAFR)*, pages 235–250, New York, NY, 2006.

[12] L. Han, L. Rudolph, J. Blumenthal, and I. Valodzin. Convexly stratified deformation spaces and efficient path planning for planar closed chains with revolute joints. In *Int. J. Robot. Res.*, pages 1189–1212, 2008.

[13] Li Han. Hybrid probabilistic roadmap — Monte Carlo motion planning for closed chain systems with spherical joints. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 920–926, New Orleans, LA, 2004.

[14] Li Han and Lee Rudolph. Inverse kinematics for a serial chain with joints under distance constraints. In *Robotics Science and Systems II*, pages 177–184. The MIT Press, Cambridge, MA, 2007. book contains the proceedings of Robotics: Science and Systems (RSS) conference, Philadelphia, PA, 2006.

[15] Li Han and Lee Rudolph. A unified geometric approach for inverse kinematics of a spatial chain with spherical joints. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 4420–4427, Roma, Italy, 2007.

[16] Li Han and Lee Rudolph. Simplex-tree based kinematics of foldable objects as multi-body systems involving loops. In *Robotics Science and Systems IV*, page to appear. The MIT Press, Cambridge, MA, 2009. book contains the proceedings of Robotics: Science and Systems (RSS) conference, Zurich, Switzerland, 2008.

[17] Li Han, Lee Rudolph, Michael Chou, Emily Eagle, Dylan Glotzae, Jake Kramer, Jonathan Moran, Christopher Pietras, Ammar Tareen, and Mathew Valko. Configurations and path planning of convex planar polygonal loops. In *Proc. Int. Workshop on Algorithmic Foundations of Robotics (WAFR)*, 2012.

[18] Li Han, Lee Rudolph, Sam Dorsey-Gordon, Dylan Glotzer, Dan Menard, Jon Moran, and James R. Wilson. Bending and kissing : Computing self-contact configurations of planar loops with revolute joints. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 1346–1351, 2009.

[19] L. Jaillet and J.M. Porta. Path planning with loop closure constraints using an atlas-based RRT. In *15th International Symposium on Robotics Research*, 2011.

[20] Marcelo Kallmann, Amaury Aubel, Tolga Abaci, and Daniel Thalmann. Planning collision-free reaching motion for interactive object manipulation and grasping. *Computer Graphics Forum*, 22(3):313–322, September 2003.

[21] L. E. Kavraki, P. Švestka, J. C. Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Trans. Robot. Automat.*, 12(4):566–580, August 1996.

[22] J.-C. Latombe. *Robot Motion Planning.* Kluwer Academic Publishers, Boston, MA, 1991.

[23] S. M. LaValle and J. J. Kuffner. Randomized kinodynamic planning. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 473–479, 1999.

[24] S. M. LaValle and J. J. Kuffner. Rapidly-exploring random trees: Progress and prospects. In *New Directions in Algorithmic and Computational Robotics*, pages 293–308. A. K. Peters, 2001. book contains the proceedings of the International Workshop on the Algorithmic Foundations of Robotics (WAFR), Hanover, NH, 2000.

[25] S.M. LaValle, J.H. Yakey, and L.E. Kavraki. A probabilistic roadmap approach for systems with closed kinematic chains. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 1671–1676, Detroit, MI, 1999.

[26] J.-M. Lien. Hybrid motion planning using minkowski sums. In *Proceedings of the Robotics: Science and Systems Confrence (RSS)*, June 2008.

[27] Barrett Technology LLC. WAM Robot, 2016. http://www.barrett.com.

[28] T. Lozano-Pérez and M. A. Wesley. An algorithm for planning collision-free paths among polyhedral obstacles. *Communications of the ACM*, 22(10):560–570, October 1979.

[29] Troy McMahon, Shawna Thomas, and Nancy M. Amato. Sampling based motion planning with reachable volumes: Application to manipulators and closed chain systems. In *Proc. IEEE Int. Conf. Intel. Rob. Syst. (IROS)*, pages 3705–3712, Chicago, Il., Sept. 2014.

[30] Troy McMahon, Shawna Thomas, and Nancy M. Amato. Sampling based motion planning with reachable volumes: Theoretical foundations. In *Proc. IEEE Int.*

Conf. Robot. Autom. (ICRA), pages 6514–6521, Hong Kong, China, May 2014.

[31] Troy McMahon, Shawna Thomas, and Nancy M. Amato. Reachable volume RRT. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 2977–2984, Seattle, Wa., May 2015.

[32] Jean-Pierre Merlet. Still a long way to go on the road for parallel mechanisms. In *ASME Bienneal Mech. Rob. Conf.*, Montreal, Canada, 2002.

[33] R. James Milgram and Jeff C. Trinkle. The geometry of configuration spaces for closed chains in two and three dimensions. *Homology, Homotopy Appl.*, 6(1):237–267, 2004.

[34] Giuseppe Oriolo and Christian Mongillo. Motion planning for mobile manipulators along given end-effector paths. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 2154–2160, Barcelona, Spain, 2005.

[35] Giuseppe Oriolo, Mauro Ottavi, and Marilena Vendittelli. Probabilistic motion planning for redundant robots along given end-effector paths. In *Proc. IEEE Int. Conf. Intel. Rob. Syst. (IROS)*, pages 1657–1662, Lausanne, Switzerland, 2002.

[36] J. H. Reif. Complexity of the mover's problem and generalizations. In *Proc. IEEE Symp. Foundations of Computer Science (FOCS)*, pages 421–427, San Juan, Puerto Rico, October 1979.

[37] Rolf Schneider. *Convex Bodies: The Brunn-Minkowski Theory.* Cambridge University Press, Cambridge, MA, 1993.

[38] Chansu Suh, Beobkyoon Kim, and Frank C. Park. The tangent bundle RRT algorithms for constrained motion planning. In *13th World Congress in Mechanism and Machine Science*, 2011.

[39] Gabriel Tanase, Antal A. Buss, Adam Fidel, Harshvardhan, Ioannis Papadopoulos, Olga Pearce, Timmie G. Smith, Nathan Thomas, Xiabing Xu, Nedal Mourad, Jeremy Vu, Mauro Bianco, Nancy M. Amato, and Lawrence Rauchwerger. 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, USA, February 12-16, 2011*, pages 235–246, 2011.

[40] Xinyu Tang, Shawna L. Thomas, Phillip Coleman, and Nancy M. Amato. Reachable distance space: Efficient sampling-based planning for spatially constrained systems. *I. J. Robotic Res.*, 29(7):916–934, 2010.

[41] Jeff C. Trinkle and R. James Milgram. Complete path planning for closed kinematic chains with spherical joints. *Int. J. Robot. Res.*, 21(9):773–789, 2002.

[42] Aimée Vargas Estrada, Jyh-Ming Lien, and Nancy M. Amato. Vizmo++: a visualization, authoring, and educational tool for motion planning. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 727–732, May 2006.

[43] Li-Chun Tommy Wang and Chih Cheng Chen. A combined optimization method for solving the inverse kinematics problem of mechanical manipulators. *IEEE Trans. Robot. Automat.*, 7(4):489–499, 1991.

[44] D. Xie and N. M. Amato. A kinematics-based probabilistic roadmap method for high DOF closed chain systems. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 473–478, New Orleans, LA, 2004.

[45] Jeffery H. Yakey, Steven M. LaValle, and Lydia E. Kavraki. Randomized path planning for linkages with closed kinematic chains. *IEEE Trans. Robot. Automat.*, 17(6):951–958, 2001.

[46] Zhenwang Yao and K. Gupta. Path planning with general end-effector constraints: using task space to guide configuration space search. In *Proc. IEEE Int. Conf. Intel. Rob. Syst. (IROS)*, pages 1875–1880, Edmonton, Alberta, Canada, 2005.

[47] A. Yershova, L. Jaillet, T. Simeon, and S. M. LaValle. Dynamic-domain RRTs: Efficient exploration by controlling the sampling domaine. In *ICRA*, pages 3867–3872, 2005.

[48] A. Yershova and S. M. LaValle. Motion planning for highly constrained spaces. In *Robot Motion and Control*, pages 297–306, 2009.

[49] Y. Zhang, K. Hauser, and J. Luo. Unbiased, scalable sampling of closed kinematic chains. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2013.

[50] M. Zucker, N. Ratli, A. D. Dragan, M. Pivtoraiko, M. Klingensmith, C. M. Dellin, J. A. Bagnell, and S. S. Srinivasa. Chomp: Covariant hamiltonian optimization for motion planning. In *The International Journal of Robotics Research*, volume 32, pages 1164–1193, 2013.