Motion Planning along Manifolds with Geodesic Convexity and Analytic Inverse Kinematics

by

Thomas B. Cohn

B.S. Honors in Mathematics, University of Michigan, 2022
B.S.E. Computer Science, University of Michigan, 2022

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2024

Authored by:	Thomas B. Cohn Department of Electrical Engineering and Computer Science May 15, 2024
Certified by:	Russell L. Tedrake Professor of Electrical Engineering and Computer Science, Thesis Supervisor
Accepted by:	Leslie A. Kolodziejski Professor of Electrical Engineering and Computer Science Chair, Department Committee on Graduate Students

Motion Planning along Manifolds with Geodesic Convexity and Analytic Inverse Kinematics

by

Thomas B. Cohn

Submitted to the Department of Electrical Engineering and Computer Science
on May 15, 2024 in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

ABSTRACT

Collision-free motion planning is a fundamental problem in robotics. Most motion planning algorithms operate in the configuration space of a robot, where each dimension corresponds to an individual degree of freedom. Oftentimes, these configuration spaces can be viewed as Euclidean spaces, and many motion planning algorithms treat them as such. However, many configuration spaces of interest are inherently non-Euclidean, including those of mobile robots, robot arms that have revolute joints without limits or ball joints, and flying robots, as well as the constrained configuration spaces that arise when planning with task-space constraints. In this thesis, we treat the problem of motion planning along Riemannian manifolds, a broader class of spaces that encompasses many of the problems of interest.

In the first chapter, we present a generalization of the graph of convex sets (GCS) planning framework that can handle smooth manifolds. GCS uses convex optimization, and is thus restricted to Euclidean configuration spaces. Our analysis utilizes geodesic convexity to achieve the same guarantees on Riemannian manifolds, and we leverage this to produce motion plans for mobile robots whose arms have unbounded revolute joints.

In the second chapter, we specifically consider the problem of constrained bimanual manipulation, where a robot has to move an object that is being grasped with two hands. The set of kinematically-valid configurations is a union of submanifolds, implicitly defined by nonlinear equality constraints This presents significant challenges for standard unconstrained planning algorithms. We construct a smooth parametrization of the feasible set, recasting the problem without equality constraints. Our approach is algorithm-agnostic, and we demonstrate that unconstrained planners (working through the parametrization) produce favorable results.

Thesis supervisor: Russell L. Tedrake
Title: Professor of Electrical Engineering and Computer Science

Acknowledgments

This thesis would not have been possible without the help of so many people. First, I want to thank my advisor, Russ Tedrake. Russ, you have been such an incredible mentor, collaborator, teacher, and friend. Getting to work together with you on this research project has been an immense privilege, and I am so excited to continue learning from you. It has also been incredibly fun, in no small part thanks to your motivation and encouragement at every step of the way.

Next, I want to thank my collaborators for their insights, efforts, and time; without them, the papers that went into this thesis quite simply would not have been written. Thank you to Mark Petersen for helping me translate our complicated math into an algorithm that we could actually implement, and for teaching me to be a better software engineer. Thank you to Seiji Shaw for the countless conversations on the theory behind these problems, and for always being there to bounce ideas back and forth. Thank you to Max Simchowitz for humoring my neuroses about the writing process, for all the time you spent helping me refine the presentation of the theoretical content, and for knowing just what to say to get me unstuck when I was struggling with the proofs in this paper. Besides my direct collaborators, I also want to thank all of my colleagues from the Robot Locomotion Group for their suggestions, ideas, and outstanding questions.

It would be remiss of me not to mention my undergraduate research advisor, Chad Jenkins. I got my start doing robotics research in his lab at the University of Michigan, and for that, I am forever grateful. I want to give a special thank you to Karthik Desingh for guiding me through writing my first paper, as well as Zhen Zeng and Nikhil Devraj for being such amazing collaborators. Thank you also to all the other members of the Laboratory for Progress, for supporting me all along the way. Also from the University of Michigan, thank you to Alejandro Uribe for being the best differential geometry professor. Your classes truly shaped how I think about mathematics, and I use the material you taught me every day in my research.

Finally, I want to thank my friends and family. Thank you to my friends from Michigan who continue to be an important part of my life, and my new friends here at MIT. Thank you to everyone from MIT Hillel, for making MIT feel like home from the moment I stepped on campus. Thank you to my parents, Amy and Jonathan, and my brother, Peter, for making me the person I am today, and supporting me at every turn. And of course, thank you to my fiancée, Molly Cooke, for your unending love, encouragement, and support.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Robot motion planning is one of the central problems in robotics. It is the question of how self-driving cars can navigate city streets, robot arms can move parts on an assembly line, or drones can fly through obstacle-rich environments. Within the famous "sense, plan, act" paradigm for autonomous robotics, motion planning is a key part of the "plan" stage: given a model of the world, as obtained from prior knowledge and sensor information, the robot must plan its motion to accomplish a given task.

If we consider the set of all possible configurations a robot could be in throughout the world (called the robot's *configuration space* [1]), then motion planning can be posed as finding a continuous path from one configuration to another. While this is a simple overall problem statement, one can consider further objectives or constraints in the problem, depending on the robot's morphology. For example, a drone flying through a forest may need to plan a path that avoids tree branches, while also respecting the constraints that result from the dynamics of flying. A robot arm on an assembly line may want to plan paths that can be traversed as quickly as possible, to maximize throughput and productivity. A self-driving car may take into account road closures, speed limits, and traffic conditions, so as to plan a faster route.

The great variety of specific motion planning problems has driven researchers to produce a rich body of literature over the past decades. One specific problem domain we will focus on in this thesis is *collision-free kinematic motion planning.*

"Kinematic" denotes the fact that we are not considering the dynamics of the robot – we consider the physical path the robot travels along, but not how fast it goes along that path, or whether such a path is even possible to follow. Such a modeling choice is appropriate for *holonomic* robots, i.e., robots which can instantaneously move in any direction. Some common examples of holonomic robots are the robot arms (such as those used in manufacturing) and wheeled *mobile manipulators*: mobile robots with arm(s) that can work in human environments, and even collaborate with humans. These types of robot systems can generally follow any path through their configuration space, although they may need to slow down if there are sharp turns.

1.1.1 Challenges Associated with Obstacle Avoidance

"Collision-free" denotes the primary challenge in this type of motion planning. For a robot working in obstacle-rich environments (including human environments), a robot should carefully avoid bumping into obstacles[1]. Checking if a hypothetical configuration of the robot is in collision with the world is straightforward [2], [3]. The three-dimensional pose in the world of each piece of the robot can be determined via *forward kinematics*, which amounts to a few matrix multiplications. From there, detecting collisions reduces to a well-studied and completely tractable geometric computation. However, it is very challenging to invert this forward kinematic mapping, so it is generally intractable to explicitly describe the obstacles in configuration space.

Fortunately, there are many approaches to plan robot motions without such explicit obstacle descriptions. One option is to extend collision-checking to compute the direction which will move the robot away from nearby obstacles. This can be used across the configuration space to try and push a candidate trajectory away from collisions. Another family of strategies relies on drawing random samples from configuration space, and checking each one for collisions. Those in collision are discarded, and those not in collision are connected into a roadmap, building up an inner-approximation of the collision-free configuration space.

Recent work [4], [5] has gone a step beyond working with individual collision-free points, instead building plans through voluminous collision-free regions of configuration space. This has gone hand-in-hand with new algorithms that can build such regions in more complex and challenging configuration spaces [6]–[8]. However, these methods have been intricately tied to *Euclidean* configuration spaces, which does not encompass all of the robots in use today.

1.1.2 Non-Euclidean Configuration Spaces

Euclidean space is a generalization of the ordinary three-dimensional space we exist in to any number of dimensions. Thus, many robot configuration spaces can naturally be interpreted as a Euclidean space. For example, the configuration space of a robot arm with n joints can be viewed as a box subset of $\mathbb{R}^n$. Each joint of the arm corresponds with one dimension of the configuration space, where the upper and lower bounds of the box correspond to the upper and lower joint limits.

However, certain robot configuration spaces are inherently non-Euclidean. Consider a mobile robot that is turning around in place. This corresponds to moving in a straight line in configuration space; when traveling along a straight line, one will never return to the same point. But after making a full revolution, the robot is right back where it started. Certain robots may have revolute joints that do not have any limits (often called *continuous* revolute joints); these joint types exhibit the same property. The set of orientations in 3D space – essential for describing the configuration of a flying or underwater robot – exhibits an even more complicated non-Euclidean structure. Furthermore, certain constraints that are often

[1]An alternative paradigm would be to move cautiously and gently, so that collisions will not damage the robot or its surroundings. The use of force or tactile sensing is essential for the robot to notice when it bumps into obstacles and modify its motion. But rigorously planning collision-free paths is crucial for fast motions.

added to motion planning problems can lead to the configuration space being non-Euclidean. For example, when two robot arms grasp a shared object, the relative pose of the hands must remain constant.

One way to describe these configuration spaces is as a *smooth manifold*, a generalization of the idea of a smooth surface to any number of dimensions. Although smooth manifolds are fundamentally more complex than the ordinary Euclidean space, many valuable properties of Euclidean space carry over. The mathematical study of smooth manifolds is called *differential geometry*, and we can leverage results from that field of study to build better motion planning algorithms.

1.2 Contributions

In this thesis, we examine the problem of collision-free kinematic motion planning for robots whose configuration spaces are smooth manifolds. First, we examine the general theory of motion planning along manifolds, building upon a state-of-the-art motion planning framework based on finding shortest paths through graphs of convex sets (GCS) in Euclidean space [9]. GCS is based on convex analysis, which is specific to Euclidean space. However, *geodesic* convexity is a natural generalization of convexity to smooth manifolds, and we use this notion to propose a Graph of *Geodesically-Convex* Sets. We present a rigorous theoretical analysis of this new framework, demonstrating that such a representation is appropriate for motion planning. These mathematical results inform simple and elegant algorithmic modifications, while preserving the guarantees of ordinary GCS. Theoretical and empirical results demonstrate the value of using our approach for mobile robots and those robots which have continuous revolute joints. We also carefully analyze the limitations of this framework, explaining how certain configuration spaces have fundamental aspects that preclude the use of our framework.

Then, we will turn our attention to the problem of constrained bimanual manipulation, where a robot moves an object that is held with both hands. The configuration space that arises as a result of this constraint is a smooth manifold. Unlike the configuration spaces of mobile robots and arms with continuous revolute joints, this configuration is implicitly described as the set of solutions to a complex system of nonlinear equations. A standard approach would be to construct an approximation of this space, but we build upon a wealth of literature on robot kinematics to describe an explicit parametrization. Acting entirely within the parametrized space, abstracts away the challenging constraints. Thus, our parametrization strategy allows general motion planning algorithms to handle this planning domain, without any specialization to handle the constraints. We empirically demonstrate that our parametrization synergizes with a variety of standard motion planning algorithms. Furthermore, the parametrization approach enables the use of planning with GCS in this domain, which previously could not be applied to such configuration spaces.

1.3 Organization

The remainder of the thesis is organized as follows. In Chapter 2, we describe the Graph of Geodesically-Convex Sets. This includes an overview of the necessary background in differential geometry in Section 2.3 and a detailed explanation of how to specialize this framework for motion planning in Section 2.6. Many of the proofs for this chapter are deferred to Appendix A..

In Chapter 3, we specifically treat the problem of planning motions for bimanual robots that are jointly grasping an object with both hands. Our parametrization relies on previous work that carefully describes the topological structure of inverse kinematics, which we summarize in Subsection 3.3.1, before presenting our explicit parametrization in Subsection 3.3.2. In Subsection 3.3.4, we describe the modifications that are needed to use the major motion planning paradigms together with our parametrization.

Finally, we conclude with a discussion of high-level takeaways in Chapter 4. Motion planning along manifolds remains an active and exciting area of research; we present multiple directions for future work building upon both the general GGCS framework (in Subsection 4.1.1) and the parametrization strategy for bimanual motion planning (in Subsection 4.1.2).

Chapter 2

Non-Euclidean Motion Planning with Graphs of Geodesically-Convex Sets

The work in this chapter was originally published in the proceedings of Robotics: Science and Systems XIX [10].

2.1 Introduction

Planning the motion of robots through environments with obstacles is a long-standing and ever-present problem in robotics. In this chapter, we aim to find the shortest path between a start and goal configuration with guaranteed collision avoidance. We are particularly motivated by planning for bimanual mobile manipulators, such as the PR2 (Willow Garage). Such robots are well-suited for a variety of tasks in human environments but present various challenges for existing motion planning algorithms.

Most popular approaches for this task fall into two categories: sampling-based planners and trajectory optimizers. The trajectory optimization problem is inherently nonconvex when there are obstacles in the scene, so solvers frequently get stuck in local minima. In that case, they may output a path that is longer than the global optimum or even fail to produce a valid path even when one exists.

On the other hand, sampling-based planners can avoid getting stuck in local minima, but the path may be locally suboptimal, resulting in jerky and uneven motion. Sampling-based planners may also suffer from the so-called "Curse of Dimensionality". Because they rely on covering the configuration space with discrete samples, in the worst case, the number of samples required may increase exponentially with the dimension. The PR2 has two 7-DoF arms and a mobile base, and sampling-based planners struggle with the instances we study here.

Recently, Marcucci, Petersen, Wrangel, *et al.* [4] described a new type of motion planning, based on a decomposition of the collision-free subset of configuration space (C-Free) into convex sets. They leverage a new optimization framework, a *Graph of Convex Sets* (GCS), where each vertex is associated with a convex set and each edge is associated with a convex function [9]. Motion planning becomes a shortest-path problem in this space. This *GCS Trajectory Optimization* approach (abbreviated as *GcsTrajOpt*) has been successfully applied

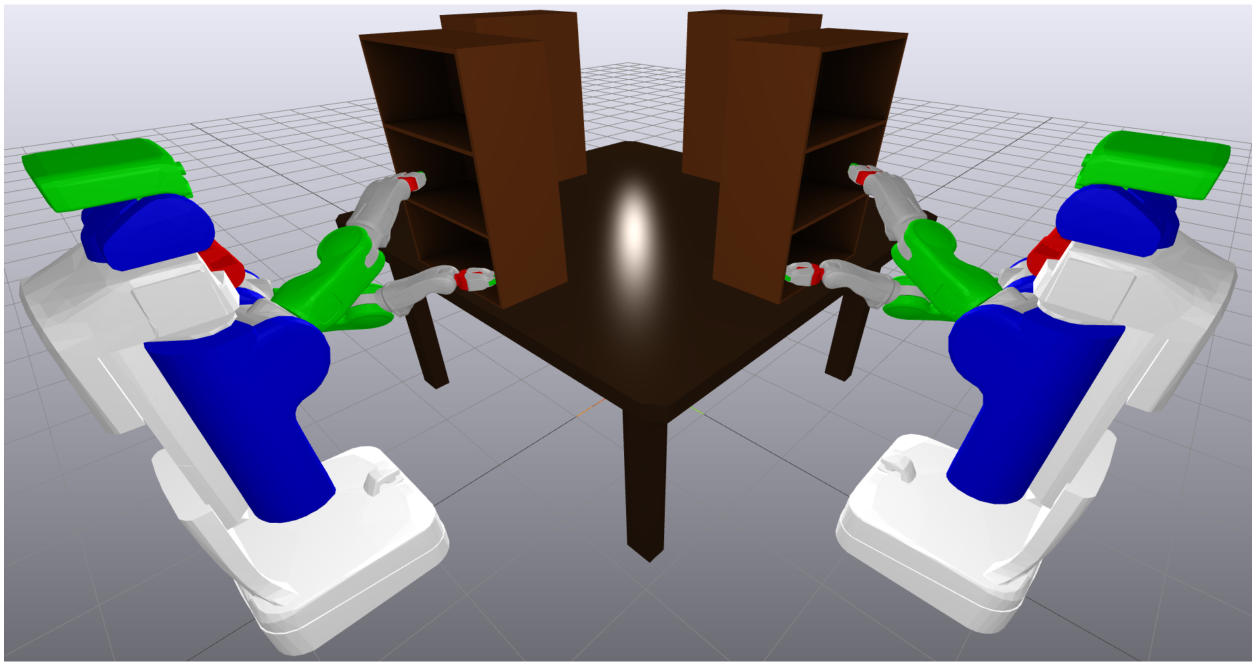


Figure 2.1: The start and goal pose for one of our motion planning experiments, using the PR2 bimanual mobile manipulator.

to challenging, high-dimensional problems, including bimanual manipulation problems.

However, GcsTrajOpt is limited to Euclidean configuration spaces. A mobile manipulator's configuration space is inherently non-Euclidean due to the mobile base: the robot can rotate through a full 360°, and its configuration is identical to when it started. Continuous revolute joints present a similar issue. Although the configuration spaces of interest are inherently non-Euclidean, they are still "locally" Euclidean, leading to elegant descriptions as differentiable manifolds. With a Riemannian metric, which allows one to measure distance on a manifold, the concepts of convexity generalize to nonlinear spaces. This in turn allows optimization on manifolds with rigorous guarantees, analogous to those obtained from convex optimization on Euclidean spaces.

In this chapter, we formulate the general problem of shortest-path motion planning around obstacles on Riemannian manifolds. We define a graph of *geodesically*-convex sets (GGCS), the analogue to GCS on a Riemannian manifold. We prove that this formulation has all the requisite properties needed to inherit the same guarantees as (Euclidean) GCS. We then turn our attention to a certain class of robot configuration spaces, encompassing open kinematic chains with continuous revolute joints and mobile bases. We show that in this case, our theoretical developments lead to simple and elegant modifications to the original GcsTrajOpt. We entitle this generalization *GGCS Trajectory Optimization* (abbreviated as *GgcsTrajOpt*), and demonstrate its efficacy with several challenging motion planning experiments.

2.2 Related Work

In the world of continuous motion planning around obstacles, most popular techniques fall into two categories: sampling-based planners and trajectory optimizers.

Sampling-based motion planners partially cover C-Free with a large number of discrete samples. Two of the foundational sampling-based planning algorithms are Probabilistic Roadmaps (PRMs) [11] and Rapidly-Exploring Random Trees (RRTs) [12]. Such algorithms are probabilistically complete, i.e., with enough samples, they will always find a valid path (if one exists). However, these algorithms are only effective if a valid plan can be produced with a reasonable number of samples. Hence, the "curse of dimensionality" is a potential obstacle to sampling-based planning, and such techniques have struggled with high-dimensional problems such as bimanual manipulation. In most cases, planning for bimanual tasks is accomplished by planning for one arm, then planning the second arm independently while treating the first arm as a dynamic obstacle. This is a reasonable heuristic for some tasks, but it sacrifices even probabilistic completeness.

An alternative approach is to formulate motion planning as an optimization problem. This requires parametrizing the space of all trajectories and defining constraints and cost functions that describe the suitability of each trajectory. Examples of kinematic trajectory optimization include B-spline parametrizations using constrained optimization [13, §7.2], CHOMP [14], STOMP [15], and KOMO [16]. Trajectory optimization approaches do not suffer from the curse of dimensionality, and are suitable for much more complex robotic systems. But the optimization landscape is inherently nonconvex, so trajectory optimization methods cannot guarantee global optimality, and often fail to produce feasible trajectories altogether.

The use of mixed integer programming (MIP) to solve motion planning problems to global optimality has recently seen an increase in popularity as new theoretical results, greater computational resources, and powerful commercial solvers [17], [18] have been brought to bear. The survey paper of Ioan, Prodan, Olaru, *et al.* [19] provides an overview of the use of MIP in motion planning. Besides the work of Marcucci, Umenberger, Parrilo, *et al.* [9], Landry, Deits, Florence, *et al.* [20] used MIP to plan aggressive quadrotor flights through obstacle-dense environments. MIP has been used to plan footstep locations for humanoid robots [21] and for quadrupeds [22], [23]. Dai, Izatt, and Tedrake [24] used MIP to globally solve the inverse kinematics problem. Finally, MIP has seen extensive use in hybrid task and motion planning [25]–[29].

Mixed integer programs can take a long time to solve in the worst case, but it is often possible to mitigate this problem with appropriate relaxations or approximations [4], [30]. GCS in particular uses an MIP formulation with a small number of integer variables, making branch-and-bound tractable. Furthermore, the convex relaxation is tight, enabling efficient approximation by solving only a convex problem combined with a randomized rounding strategy. [4] argued that for single-arm manipulators, this approach can find more optimal plans in less time than PRMs. These valuable properties carry over to our extension of GCS.

Another recent trend in motion planning has been the use of Riemannian geometry to model the problem. Riemannian Motion Policies (RMPs) [31] combine acceleration-based controllers across different task spaces into a single unified controller. A Riemannian metric in each task space determines the priority of a given controller, and smooth maps between the manifolds enable the averaging of controllers. RMPs have seen continued improvement [32], [33] and generalization [34], [35]. Klein, Jaquier, Meixner, *et al.* [36] envision Riemannian geometry as a tool for generating and combining elegant motion synergies for complex robotic systems.

2.3 Preliminaries

In this section, we cover some of the relevant mathematical background. We supply intuitive definitions; for further reference on Riemannian geometry, see the textbooks of Do Carmo [37] and Lee [38], [39]. Boumal [40] provides an excellent treatment of optimization over manifolds. We use the notation $[n] = \{1, \ldots, n\}$.

2.3.1 Riemannian Geometry

A d-dimensional (topological) *manifold* $\mathcal{M}$ is a locally Euclidean topological space: for any $p \in \mathcal{M}$, there is an open neighborhood $\mathcal{U}$ of p and a continuous map $\psi : \mathcal{U} \to \mathbb{R}^d$ which is a homeomorphism onto its image. The pair $(\mathcal{U}, \psi_{\mathcal{U}})$ is called a *coordinate chart*, and for any pair of overlapping charts $(\mathcal{U}, \psi_{\mathcal{U}})$ and $(\mathcal{V}, \psi_{\mathcal{V}})$, we have a *transition* map

$$\tau_{\mathcal{U},\mathcal{V}} = \psi_{\mathcal{V}} \circ \psi_{\mathcal{U}}^{-1}\big|_{\psi_{\mathcal{U}}(\mathcal{U}\cap\mathcal{V})} \tag{2.1}$$

A collection of charts whose domains cover the manifold is called an *atlas*. We only consider $\mathcal{C}^\infty$-*smooth* manifolds, where all transition maps in the atlas are $\mathcal{C}^\infty$.

For each $p \in \mathcal{M}$, the *tangent space* $\mathcal{T}_p\mathcal{M}$ is a d-dimensional vector space representing the set of directional derivatives at p. Given a differentiable curve $\gamma : (-\epsilon, \epsilon) \to \mathcal{M}$ with $p = \gamma(0)$, this affords an interpretation of the *velocity* of γ at p, $\gamma'(0)$, as an element of $\mathcal{T}_p\mathcal{M}$. For a smooth map of manifolds $f : \mathcal{M} \to \mathcal{N}$, the *pushforward* of f at p is a linear map $f_{*,p} : \mathcal{T}_p\mathcal{M} \to \mathcal{T}_{f(p)}\mathcal{N}$, generalizing the Jacobian matrix [38, page 55]. The pushforward is defined so that, with γ defined as above, $f_{*,p}(\gamma'(0)) = (f \circ \gamma)'(0)$.

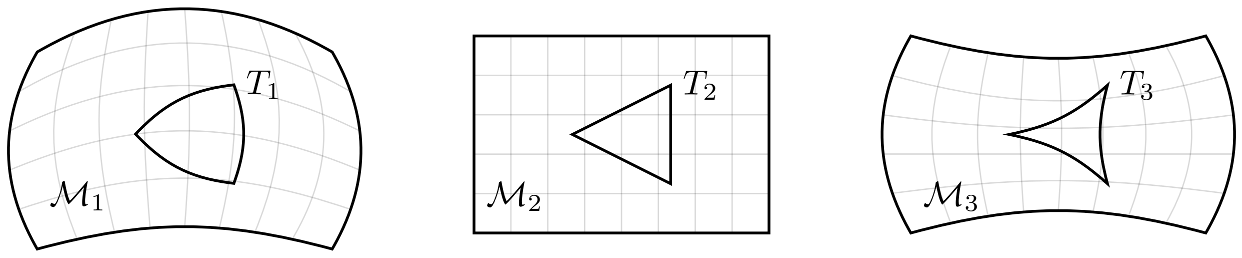
A *Riemannian metric* g is a smoothly-varying positive-definite bilinear form over $\mathcal{M}$ that gives each tangent space $\mathcal{T}_p\mathcal{M}$ an inner product $\langle \cdot\,, \cdot \rangle_p^{(\mathcal{M},g)}$. The pair $(\mathcal{M}, g)$ is a *Riemannian manifold*, and we frequently refer to $\mathcal{M}$ exclusively when the choice of metric is unambiguous. A Riemannian metric allows one to measure the length of a curve, invariant to reparametrizations [39, page 34]; if $\gamma : [a, b] \to \mathcal{M}$ is piecewise continuously differentiable, then

$$L(\gamma) = \int_a^b \sqrt{\langle \gamma'(s), \gamma'(s) \rangle_{\gamma(s)}^{(\mathcal{M},g)}}\, \mathrm{d}s \tag{2.2}$$

We call the integrand the *speed* of γ. The distance between any two points $p, q \in \mathcal{M}$ is defined as the infimum of the arc length of all curves connecting them:

$$\mathsf{d}_{\mathcal{M}}(p, q) = \inf \left\{ L(\gamma) \middle| \gamma \in \mathcal{C}^1_{\mathrm{pw}}([0, 1], \mathcal{M}), \gamma(0) = p, \gamma(1) = q \right\} \tag{2.3}$$

where $\mathcal{C}^1_{\mathrm{pw}}([0, 1], \mathcal{M})$ is the set of parametric piecewise-continuously differentiable curves from the interval $[0, 1]$ to $\mathcal{M}$. A curve that achieves this infimum need not exist in general [37,



(a) Positive (b) Zero (flat) (c) Negative

Figure 2.2: Examples of geodesic triangles T_i in manifolds $\mathcal{M}_i$ with various sectional curvatures. In positive curvature spaces, the interior angles sum to more than 180°, and the edges bow outwards. The opposite is true in negative curvature spaces.

page 146]. We also define $\mathsf{d}_{\mathcal{U}}(p,q)$ for $p,q \in \mathcal{U} \subseteq \mathcal{M}$ to be the infimum of the length of paths whose image is contained in $\mathcal{U}$.

If $\mathcal{M}$ is connected, it is a metric space with respect to $\mathsf{d}_{\mathcal{M}}$. Given two Riemannian manifolds $(\mathcal{M},g)$ and $(\mathcal{N},h)$, a smooth function $f:\mathcal{M}\to\mathcal{N}$ is a *local isometry* if

$$\langle u,v\rangle_p^{(\mathcal{M},g)} = \langle f_{*,p}(u), f_{*,p}(v)\rangle_{f(p)}^{(\mathcal{N},h)} \tag{2.4}$$

$\forall p\in\mathcal{M}$, $\forall u,v\in\mathcal{T}_p\mathcal{M}$. If f is also a diffeomorphism, and $\mathcal{M}$ and $\mathcal{N}$ are connected, then f preserves distances [39, page 37], and is an *isometry* of metric spaces. The converse is also true [41].

A *geodesic* is a locally length-minimizing curve, parameterized to be constant speed. Locally length-minimizing means that for two points on the geodesic that are close enough, the geodesic traces out the shortest path between them. For example, geodesics in Euclidean space with the natural metric are lines, rays, and line segments, and geodesics on the sphere (with the induced metric from Euclidean space) are great circles. Constructing the shortest geodesic between two points is a variational calculus problem, so the solution must satisfy the Euler-Lagrange system of differential equations. Thus, initial conditions $p\in\mathcal{M}$ and $v\in\mathcal{T}_p\mathcal{M}$ uniquely define a geodesic, such that v is the velocity of the geodesic as it passes through p. This is used to define the *exponential map* $\exp_p:\mathcal{T}_p\mathcal{M}\to\mathcal{M}$, where the direction of a vector v defines the direction of the geodesic, and the magnitude of v determines how far to move in that direction away from p.

A Riemannian metric induces *curvature* on a manifold, capturing how local geometry differs from the standard Euclidean case. The *sectional curvature* at a point p is a real-valued function defined on 2-dimensional subspaces of the tangent space $\mathcal{T}_p\mathcal{M}$ [37, §4.3]. (We write $\mathcal{K}(u,v)$ for any vectors u and v that span the subspace.) Informally, the sectional curvature corresponds to the distortion of angles in triangles, as shown in Fig. 2.2. Manifolds that have everywhere-zero curvature are called *flat*, and are locally isometric to Euclidean space.

The Cartesian product of two Riemannian manifolds is itself a Riemannian manifold. The curvature of the component manifolds influences the curvature of the product. Importantly, *the product of flat manifolds is flat* [42]. As we explain in Section 2.5, this implies that a robot with a mobile base and (potentially many) continuous revolute joints has a flat configuration space.

The curvature is a fourth-order tensor with complex symmetries, but the derived *sectional curvature* (defined in terms of 2-dimensional subsets of the tangent space) provides a much more intuitive interpretation. A region of positive curvature locally looks similar to a sphere, whereas a region of negative curvature locally looks similar to a saddle.

2.3.2 Convex Analysis on Manifolds

To define convexity on a Riemannian manifold $(\mathcal{M}, g)$, we replace the notion of lines with geodesics. In general, there is not a unique geodesic (or even a unique shortest geodesic) between two points, so a more intricate definition is required. A subset $\mathcal{U} \subseteq \mathcal{M}$ is *strongly geodesically convex* (or *g-convex*) if $\forall p, q \in \mathcal{U}$, there is a unique length-minimizing geodesic connecting p and q, and it is entirely contained in $\mathcal{U}$. This definition ensures that the intersection of g-convex sets is g-convex, and that there is a unique shortest path between any pair of points in a g-convex set. Weaker definitions used in other works [43], [44] do not provide these guarantees. See [40, §11.3] for further discussion.

G-convex neighborhoods exist around every point [37, page 77] For any $p \in \mathcal{M}$, there is a *convexity radius* $r_p > 0$, such that the open ball

$$B_r(p) = \left\{ \exp_p(q) \;\middle|\; q \in \mathcal{T}_p\mathcal{M}, ||q|| < r \right\} \tag{2.5}$$

is g-convex for any $r < r_p$ (where the norm is induced by the Riemannian metric). Intuitively, the convexity radius quantifies how large a set can be before minimizing geodesics can go "the wrong way around" the manifold. On the product of two Riemannian manifolds, each geodesic is naturally the product of geodesics on its components. Thus, the product of g-convex sets is g-convex in the product manifold.

A function $f : \mathcal{M} \to \mathbb{R}$ is said to be *geodesically convex* (*g-convex*) on $\mathcal{U} \subseteq \mathcal{M}$ if, for any geodesic $\gamma : [0, 1] \to \mathcal{U}$, $(f \circ \gamma)$ is a convex function on $[0, 1]$. That is, $\forall t \in [0, 1]$,

$$f(\gamma(t)) \leq (1 - t)f(\gamma(0)) + tf(\gamma(1)) \tag{2.6}$$

We say that f is *locally* g-convex if for any $p \in \mathcal{M}$, there exists a neighborhood $\mathcal{U}_p$ of p such that the restriction of f to $\mathcal{U}$ is g-convex.

Unfortunately, existing research into g-convex optimization often focuses on specific classes of manifolds that do not encompass the configuration spaces of interest [44], [45]. In addition, there is little existing literature studying mixed-integer Riemannian convex optimization, and techniques commonly used in the Euclidean case (e.g., cutting planes [46]) may not generalize to Riemannian manifolds.

2.4 Problem Statement

We may now precisely state our kinematic planning problem in the language of Riemannian geometry developed thus far. Let $(\mathcal{Q}, g)$ be the configuration space of a robot, realized as a connected Riemannian manifold, and further assume that $\mathcal{Q}$ is *complete* [38, page 598] with respect to the metric induced by g. Suppose that the set of collision-free configurations is a bounded open subset $\mathcal{M} \subseteq \mathcal{Q}$, and without loss of generality, assume that $\mathcal{M}$ is path-connected. (If $\mathcal{M}$ is not path-connected, then we restrict ourselves to planning within a single connected component.)

Suppose we want to find the shortest path between two points p and q in $\overline{\mathcal{M}}$, the closure of $\mathcal{M}$ (i.e., the smallest closed set containing $\mathcal{M}$). This can be written as the optimization problem

$$\begin{array}{rl} \operatorname{argmin} & L(\gamma) \\ \text{s.t.} & \gamma \in \mathcal{C}^1_{\text{pw}}([0,1], \overline{\mathcal{M}}) \\ & \gamma(0) = p \\ & \gamma(1) = q \end{array} \tag{2.7}$$

where L is the Riemannian arc length, given in Eq. (2.2). In the following sections, we develop machinery to solve optimization problems of this form.

2.5 Graphs of Geodesically-Convex Sets

We now introduce a *graph of geodesically convex sets* (GGCS), a Riemannian optimization framework that, in Section 2.6, we show is general enough to encompass Problem (2.7). A GGCS is a directed graph $G = (V, E)$ with certain properties, designed as a generalization of ordinary (Euclidean) GCS from Marcucci, Umenberger, Parrilo, *et al.* [9, §2] to Riemannian manifolds. Each vertex $v \in V$ has a corresponding g-convex subset $\mathcal{Y}_v$ of some Riemannian manifold $(\mathcal{M}_v, g_v)$. With each edge $e = (u, v) \in E$, we associate a cost function $\ell_e^{\mathcal{Y}} : \mathcal{Y}_u \times \mathcal{Y}_v \to \mathbb{R}_{\geq 0} \cup \{\infty\}$, which must be g-convex with respect to the product metric on $\mathcal{M}_u \times \mathcal{M}_v$. For all problems considered in this chapter, every g-convex set will be a subset of a single Riemannian manifold.

Given distinct source and target vertices $p, q \in V$, a *path* π from p to q is a sequence of distinct vertices $(v_k)_{k=0}^K$ such that $v_0 = p$, $v_K = q$, and $(v_{k-1}, v_k) \in E$ for all $k \in [K]$. Associate to this path a sequence of points $y_\pi = (y_0, \ldots, y_K)$ such that each $y_v \in \mathcal{Y}_v$; then the length of this path is

$$\ell_\pi^{\mathcal{Y}}(y_\pi) = \sum_{k=1}^K \ell^{\mathcal{Y}}_{(v_{k-1}, v_k)}(y_{k-1}, y_k) \tag{2.8}$$

Let Π denote the set of all paths from p to q, and for any $\pi \in \Pi$, define its feasible vertices as $\mathcal{Y}_\pi = \mathcal{Y}_{v_0} \times \cdots \times \mathcal{Y}_{v_K}$. The problem of finding the shortest path from p to q can be written as

$$\min_{\pi \in \Pi} \min_{y_\pi \in \mathcal{Y}_\pi} \ell_\pi^{\mathcal{Y}}(y_\pi) \tag{2.9}$$

Solving Problem (2.9) to optimality is intractable in complete generality, so we propose to transform it into an ordinary GCS problem. To each $v \in V$, associate a chart ψ_v, and define $\mathcal{X}_v = \psi_v(\mathcal{Y}_v)$. For each edge $e = (u, v) \in E$, we define the edge cost on $(x_u, x_v) \in \mathcal{X}_u \times \mathcal{X}_v$:

$$\ell_e(x_u, x_v) = \ell_e^{\mathcal{Y}}(\psi_u^{-1}(x_u), \psi_v^{-1}(x_v)). \tag{2.10}$$

This construction is shown in Fig. 2.3. To apply the GCS machinery, we require that the sets $\mathcal{X}_v$ and edge costs $\ell_e(x_u, x_v)$ are convex. As we show in Subsection 2.6.5, this is hopeless

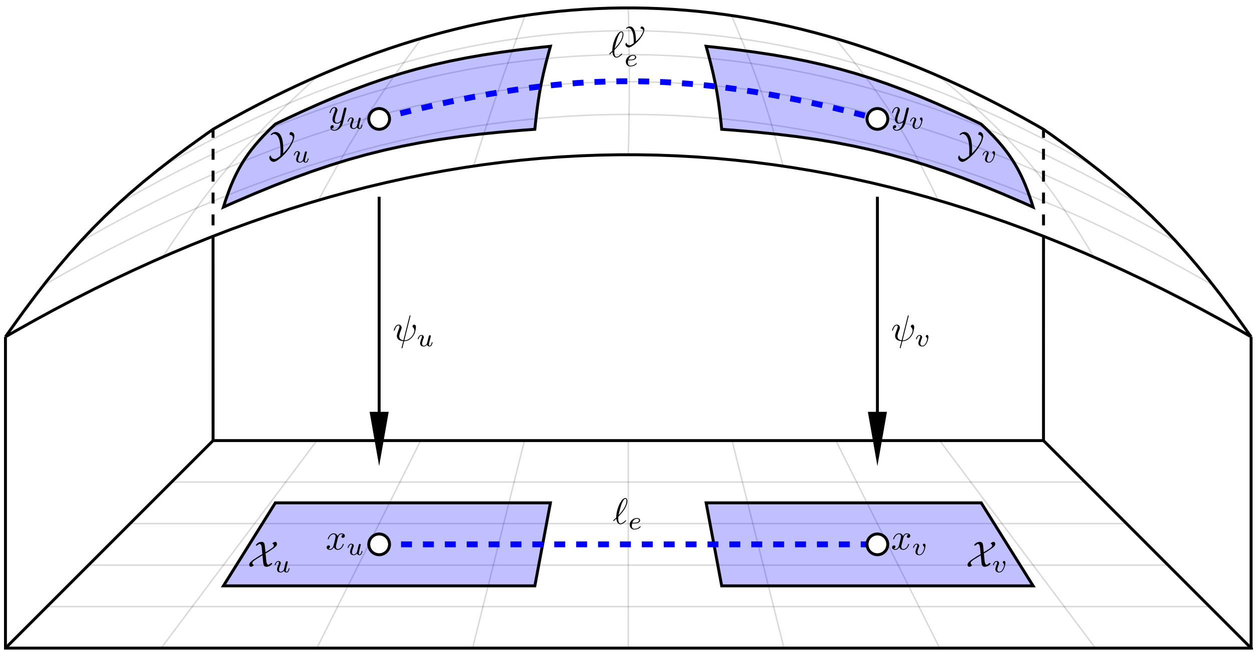


Figure 2.3: Moving edges and sets from Riemannian manifolds to Euclidean spaces with coordinate charts. In this diagram, $\mathcal{Y}_u$ and $\mathcal{Y}_v$ are visualized as part of the same Riemannian manifold, although this need not be true in general.

for manifolds with positive curvature. Luckily, for flat manifolds, convexity can be ensured, as will be shown in Subsection 2.6.1.

Importantly, many robot configuration spaces can be realized as flat manifolds. SE(2) is flat, all 1-dimensional manifolds are flat [39, page 47] (this encompasses continuous revolute joints), and products of flat manifolds are flat. Thus, any robotic system whose configuration can be described using a series of single-degree-of-freedom joints (and potentially a mobile base) will have a flat configuration space, and thus can be handled by our methodology. 2-DoF universal joints can also be handled, as they can be perfectly represented as two juxtaposed 1-DoF joints. 3-DoF ball joints cannot be handled perfectly, because decomposing a ball joint into 1-DoF joints distorts the underlying geometry. Instead, one can use a piecewise-linear approximation of this configuration space – see Subsection 2.6.5 for further discussion.

2.6 Motion Planning with GGCS

We want to use GGCS to make motion plans on Riemannian manifolds by solving Problem (2.7). Thus, we must prove that the optimal value is achieved by some trajectory that is feasible for a GGCS problem. We use the initialism ROSC (Riemannian Open Subset Closure) to describe closures of open subsets of Riemannian manifolds, notably $\overline{\mathcal{M}}$. ROSCs are topological manifolds-with-boundary, but the boundary may not be smooth; for example, polytopic obstacles lead to corners on the boundary of $\overline{\mathcal{M}}$. The theory of manifolds-with-corners is not well developed in full generality, so for the sake of completeness, we confirm some expected properties of paths through ROSCs.

Theorem 1. ***(Existence of Optimal Trajectories)*** *For any $p, q \in \overline{\mathcal{M}}$, there exists a continuous curve γ connecting them such that $L(\gamma) = \mathsf{d}(p, q)$.*

Proof. The proof follows by verifying that $\overline{\mathcal{M}}$ satisfies the preconditions of Theorem 2.5.23 of Burago, Burago, and Ivanov [47, page 50]. We defer the details to Appendix A.1. □

Assumption 1. *We are given a finite atlas $\mathcal{A} = \{(\mathcal{Y}_v, \psi_v)\}_{v \in V}$ of $\mathcal{M}$. For each v, the closure $\overline{\mathcal{Y}}_v$ is g-convex as a subset of $\mathcal{Q}$. Furthermore, the union of the closures $\overline{\mathcal{Y}}_v$ covers $\overline{\mathcal{M}}$.*

These requirements will not hold in general, but we will discuss how to construct such an atlas in Subsection 2.6.2. We can also extend each ψ_v to be defined on $\overline{\mathcal{Y}}$. With this information, we can prove a strong result about the shortest paths in $\overline{\mathcal{M}}$.

Theorem 2. ***(Piecewise Geodesic Optimal Paths)*** *Let $p, q \in \overline{\mathcal{M}}$, and suppose the sets $\overline{\mathcal{Y}}_v$ satisfy Assumption 1. Then there is a curve $\gamma^* \in \mathcal{C}^1_{\mathrm{pw}}([a, b], \overline{\mathcal{M}})$ connecting them, such that the following are true:*

- *$L(\gamma^*) = \mathsf{d}(p, q)$*
- *γ^* is a piecewise geodesic of $\mathcal{Q}$*
- *Each geodesic segment is contained in some $\overline{\mathcal{Y}}_v$*
- *γ^* passes through each $\overline{\mathcal{Y}}_v$ at most once.*

Proof. Let γ_0 be a continuous minimizing path connecting p to q (guaranteed to exist by Theorem 1); we will use this to construct an appropriate γ^*. Select an arbitrary order $v_1, \ldots, v_{|V|}$ to iterate over all of the vertices in V. We will construct a sequence of curves $\gamma_1, \ldots, \gamma_{|V|}$, such that $\gamma_{|V|}$ has the desired properties.

For each i, if γ_{i-1} does not pass through $\overline{\mathcal{Y}}_{v_i}$, let $\gamma_i = \gamma_{i-1}$. Otherwise, let $T_i = \left\{t \;\middle|\; \gamma_{i-1}(t) \in \overline{\mathcal{Y}}_{v_i}\right\}$, let $a'_i = \min(T_i)$, and let $b'_i = \max(T_i)$. Then by the g-convexity of $\overline{\mathcal{Y}}_{v_i}$, there is a unique minimizing geodesic $\alpha_i : [a'_i, b'_i] \to \overline{\mathcal{Y}}_{v_i}$ connecting $\gamma_{i-1}(a'_i)$ and $\gamma_{i-1}(b'_i)$. Let $\tilde{\gamma}$ be a new curve, defined by

$$\tilde{\gamma}(t) = \begin{cases} \gamma_{i-1}(t) & t \notin [a'_i, b'_i] \\ \alpha_i(t) & t \in [a'_i, b'_i] \end{cases} \tag{2.11}$$

Because $L(\alpha_i) \leq L(\gamma_{i-1}|_{[a',b']})$, we have $L(\tilde{\gamma}) \leq L(\gamma_{i-1})$, and since γ_{i-1} is of minimum length, we must have $L(\tilde{\gamma}) = L(\gamma_{i-1})$. Define γ_i to be $\tilde{\gamma}$, and continue until we have iterated over all of the $v \in V$. Then by construction, $L(\gamma_{|V|}) = \mathsf{d}(p, q)$, $\gamma_{|V|}$ is piecewise geodesic in $\mathcal{Q}$, each geodesic segment is contained in some $\overline{\mathcal{Y}}_v$, and $\gamma_{|V|}$ passes through each $\overline{\mathcal{Y}}_v$ at most once. □

2.6.1 Formulation as a GCS Problem

To transform the GGCS problem into a GCS problem, we require that the sets and edge costs are convex in Euclidean space. The following is sufficient (and still encompasses robots with mobile bases and continuous revolute joints):

Assumption 2. *$\mathcal{Q}$ is flat. Also, each ψ_v is a local isometry into Euclidean space, viewed as a Riemannian manifold with the canonical Euclidean metric.*

Assumptions 1 and 2 together yield three important results:

- $\mathcal{X}_v = \psi_v(\overline{\mathcal{Y}}_v)$ is convex.
- $\forall y_0, y_1 \in \overline{\mathcal{Y}}_v$, $\mathsf{d}(y_0, y_1) = ||\psi_v(y_0) - \psi_v(y_1)||_2$
- $\tau_{u,v}$ is a Euclidean isometry (see Lemma 4 in Appendix A.2), and hence affine [48].

The first two results are true because $\mathcal{Y}_v$ is g-convex, ψ_v maps geodesics to geodesics [39, page 125], and geodesics are unique in Euclidean space. For most robotic configuration spaces we consider, $\mathcal{Q}$ can be decomposed as the product of one-dimensional manifolds. In this case, the coordinate systems can be globally aligned, so that every transition map is a translation.

To formulate the problem with GCS, we follow an approach similar to [4], where decision variables describe line segments contained within each convex set. In particular, $\forall v \in V$, we have $x_v = (x_{v,0}, x_{v,1}) \in \mathcal{X}_v^2$, where $x_{v,0}$ is the start point of the line segment, and $x_{v,1}$ is the end point. For each edge $e = (u, v) \in E$, the length of the segment associated with the starting vertex is used as the edge cost:

$$\ell_e(x_u, x_v) = \mathsf{d}(\psi_u^{-1}(x_{u,0}), \psi_u^{-1}(x_{u,1})) = ||x_{u,0} - x_{u,1}||_2 \tag{2.12}$$

We also encode equality constraints to ensure the endpoints of adjacent segments are in agreement:

$$\psi_u^{-1}(x_{u,1}) = \psi_v^{-1}(x_{v,0}) \quad \Leftrightarrow \quad \tau_{u,v}(x_{u,1}) = x_{v,0} \tag{2.13}$$

This constraint is convex because $\tau_{u,v}$ is affine. Thus, we have a valid GCS formulation, which can be solved as a mixed-integer convex program. Alternatively, it can be solved approximately by solving the convex relaxation and using a randomized rounding strategy [4]. If we have $p \in \mathcal{X}_0$ and $q \in \mathcal{X}_K$, then after solving the GCS problem, we obtain a path

$$x_\pi = (x_{0,0}, x_{0,1}, x_{1,0}, x_{1,1}, \ldots, x_{K,0}, x_{K,1}) \tag{2.14}$$

with $x_{0,0} = \psi_0(p)$, $x_{K,1} = \psi_K(q)$, and $\psi_i(x_{i,1}) = \psi_{i+1}(x_{i+1,0})$, $\forall i \in \{1, \ldots, K-1\}$. Such a path naturally lifts to a path on $\overline{\mathcal{M}}$:

$$\begin{aligned} y_\pi &= (y_0 = p, y_1, y_2, \ldots, y_K, y_{K+1} = q) \\ &= (\psi_0^{-1}(x_{0,0}), \psi_1^{-1}(x_{1,0}), \ldots, \psi_K^{-1}(x_{K,0}), \psi_K^{-1}(x_{K,1})) \end{aligned} \tag{2.15}$$

where we have removed duplicate points from the trajectory. This process is visualized for a simple cylinder manifold in Fig. 2.4.

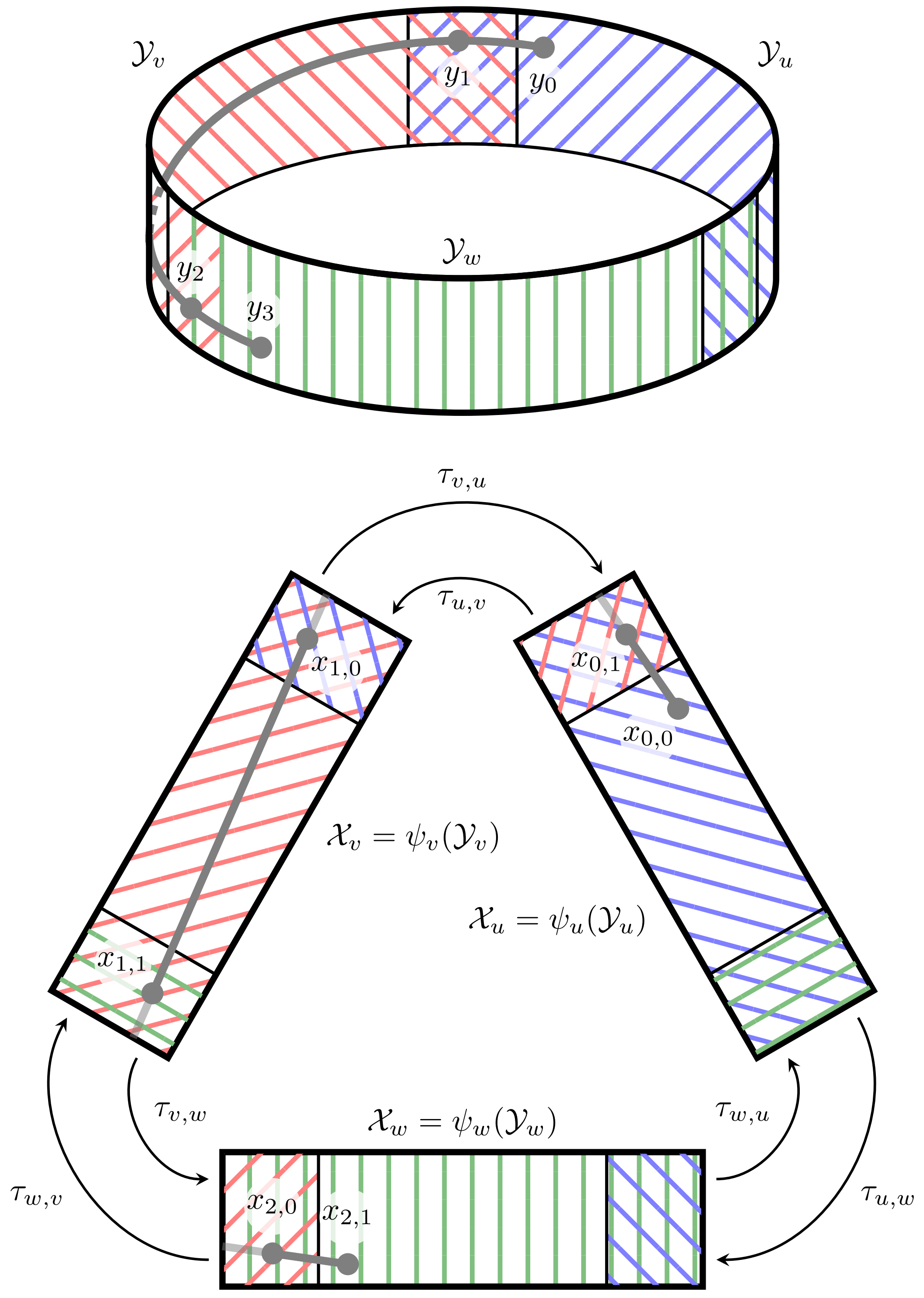


Figure 2.4: The process of transforming a GGCS problem into a GCS problem for a simple cylinder manifold. Each of the three charts maps to a Euclidean space, with transition maps encoding the equality constraints across chart domains. The line segments then lift to a piecewise geodesic on the manifold.

For each $i \in \{0, \ldots, K\}$, y_i and y_{i+1} are contained in a g-convex set $\overline{\mathcal{Y}}_i$, so there is a unique minimizing geodesic γ_i connecting them and completely contained in $\overline{\mathcal{Y}}_i$. Thus, a path y_π uniquely defines a piecewise geodesic γ_π connecting p to q that is completely contained in $\overline{\mathcal{M}}$. With this fact, we can formally prove the equivalence of the GGCS problem and the GCS problem.

Theorem 3. ***(Proof of Problem Equivalence)*** *If the path x_π given in Eq. (2.14) is optimal for the GCS problem defined by Eqs. (2.12) and (2.13), then the piecewise geodesic γ_π defined by Eq. (2.15) is optimal for Problem (2.7).*

Proof. Any feasible path x_π for the GCS problem yields a piecewise continuously differentiable curve γ_π whose image is contained in $\overline{\mathcal{M}}$ and connecting p to q. Then the length of this curve satisfies

$$L(\gamma_\pi) = \sum_{i=0}^{K} \mathsf{d}(y_i, y_{i+1}) = \sum_{i=0}^{K} ||x_{i,0} - x_{i,1}||_2 = \ell_\pi(x_\pi)$$

Thus, the optimal value of Problem (2.7) is no worse than the optimal value of the GCS problem.

Now, consider an optimal γ^* for Problem (2.7), with the properties of Theorem 2. Then γ^* is the concatenation of geodesics $\gamma_1, \ldots, \gamma_K$, where $\gamma_i : [0,1] \to \overline{\mathcal{Y}}_{v_i}$ for $i = 1, \ldots, K$, and each v_i is distinct. Define x_π by

$$(x_{i,0}, x_{i,1}) = (\psi_i(\gamma_i(0)), \psi_i(\gamma_i(1)))$$

$\forall i \in [K]$. By construction, $\ell_\pi(x_\pi) = L(\gamma^*)$. $\gamma_i(1) = \gamma_{i+1}(0)$, and the v_i are distinct, so x_π is feasible for the GCS problem. Thus, the GCS problem achieves the optimal value of Problem (2.7). □

2.6.2 Construction of the Atlas

A key part of motion planning with GGCS is the construction of an appropriate atlas $\mathcal{A} = \{(\mathcal{Y}_v, \psi_v)\}_{v \in V}$ of $\mathcal{M}$. Recall that $\mathcal{A}$ must be finite, each $\overline{\mathcal{Y}}_v$ must be g-convex, and each ψ_v must be a local isometry.

As was done in [4], we construct an inner approximation of C-Free using the extension of IRIS [7, Alg. 2] to handle nonconvex obstacles. Given a seed point in $\mathcal{Q}$, we grow a region about that point with respect to a local coordinate system. In this way, we grow the region in Euclidean space, but we have an implicit mapping to the manifold, allowing us to construct the transition maps needed for Eq. (2.13).

To ensure the set is g-convex when lifted to $\mathcal{Q}$, we bound the region by the convexity radius on a per-joint basis. If r_i is the convexity radius of the ith joint's configuration space, we constrain that joint to take values within an open ball of radius r_i, centered at the seed point. For revolute joints without limits and mobile bases, the convexity radius is $\pi/2$. (Computationally, we use a closed ball of radius $r_i - \epsilon$, with a small $\epsilon > 0$.) For a 1DoF joint, this is just the interval $[x - r_i + \epsilon, x + r_i - \epsilon]$ for seed point x. If the manifold is flat, this guarantees g-convexity (see proof in Appendix A.2).

Theorem 4. *Suppose $\mathcal{Q} = \mathcal{Q}_1 \times \cdots \times \mathcal{Q}_m$, where each $\mathcal{Q}_i$ has a convexity radius r_i. Let $(\mathcal{Y}, \psi)$ be a coordinate chart, with ψ a local isometry and $\psi(\mathcal{Y})$ convex in Euclidean space. If $\mathcal{Q}$ is flat and the diameter of $\operatorname{proj}_{\mathcal{Q}_i}(\mathcal{Y})$ is less than $2r_i$, then $\mathcal{Y}$ is g-convex.*

We also assumed full coverage of $\overline{\mathcal{M}}$ by the union of the $\mathcal{Y}_v$. In scenarios where we only have an inner approximation of C-Free, we treat all points outside of that approximation as obstacles. Thus, our planner finds the globally optimal path within "certified" C-Free, which is a subset of the whole C-Free.

2.6.3 Additional Costs and Constraints

Marcucci, Petersen, Wrangel, *et al.* [4] extended GcsTrajOpt to parametrize trajectories as piecewise Bézier curves, in order to handle a greater variety of costs and constraints. This includes penalizing the duration and energy of a trajectory, adding velocity bounds, and requiring the trajectory to be differentiable a certain number of times. Bézier curves generalize naturally to Riemannian manifolds by interpolating along the minimizing geodesics between control points [49], [50]. Because we restrict ourselves to flat manifolds, the local geometry is unchanged from Euclidean space. Thus, all costs and constraints that operate on individual segments of the piecewise Bézier curve trajectory can be used with no changes.

To enforce the differentiability of the overall trajectory where two segments connect, we must compare tangent vectors across different coordinate systems. In particular, suppose we need differentiability η times for an edge (i, j), with transition map $\tau_{i,j}$. Let γ_i and γ_j be adjoining Bézier curve segments in $\mathcal{Y}_i$ and $\mathcal{Y}_j$, and let their kth derivatives be $\nu_i^{(k)}$ and $\nu_j^{(k)}$ at the point where they connect, called w. Using the pushforward of the transition map at w, this constraint can be written as

$$(\tau_{i,j})_{*,\psi_i^{-1}(w)} \left(\nu_i^{(k)}\right) = \nu_j^{(k)} \qquad \forall l \in [\eta] \tag{2.16}$$

Because the transition map is a Euclidean isometry, its pushforward is a linear transformation described by an orthogonal matrix, and if the coordinate systems are globally aligned (as described in Subsection 2.6.1), then the pushforward is the identity map. When $\mathcal{Q}$ is flat, the derivative of a Bézier curve is a linear expression of its control points, so Eq. (2.16) is a convex constraint.

2.6.4 Beyond Flat Manifolds

The guarantees afforded by GgcsTrajOpt derive from Assumptions 1 and 2. Assumption 1 affects the completeness and optimality of the algorithm, and it may be impossible to construct an appropriate atlas if the boundary of $\overline{\mathcal{M}}$ is not a piecewise-totally-geodesic submanifold. However, we can always construct a finite atlas of g-convex sets to cover all but an arbitrarily small subset of $\overline{\mathcal{M}}$. Indeed, in practice, we simply work with an inner approximation of $\overline{\mathcal{M}}$, and GgcsTrajOpt will find the shortest path contained within that inner approximation.

Assumption 2 is used to guarantee that the resulting optimization problem is convex; without it, we may have nonconvex costs or constraints, and can make no guarantees of finding an optimal (or even feasible) solution. But many manifolds of interest in robotics do not satisfy these requirements. Certain configuration spaces are inherently not flat manifolds. Examples include SO(3), which is the configuration space of a ball joint and SE(3), which is the configuration space of a free rigid body. Planning problems where general kinematic or dynamic constraints have been imposed can also yield a constrained configuration space as an embedded non-flat submanifold of the full configuration space.

Piecewise-Linear Approximations

To handle arbitrary manifolds, we turn our attention to piecewise-linear (PL) approximations. In particular, we consider a triangulation of the manifold: a simplicial (or polytopic) mesh of appropriate dimension whose topology matches the manifold. The GCS machinery can be used to plan along a PL approximation; indeed, the original GCS paper directly considers piecewise-affine systems [9, §8.2]. In the context of approximating a smooth manifold, we will treat each simplex as a chart domain, replace the transition map with the mapping between two adjacent simplices, and approximate the arc length on the manifold with the arc length along the PL approximation.

We consider a similar problem setup to [51]. The configuration space $\mathcal{Q}$ is a d-dimensional embedded submanifold of $\mathbb{R}^n$, defined as the zero level set of a smooth function $F : \mathbb{R}^n \to \mathbb{R}^{n-d}$ whose Jacobian is full rank everywhere. We are given a piecewise-linear approximation of $\mathcal{Q}$ composed of convex polytopes $\mathcal{P}_i \subseteq \mathbb{R}^n$. For each polytope $\mathcal{P}_i$, the orthogonal projection $\psi_i : \mathcal{M} \to \operatorname{aff}(\mathcal{P}_i)$ (where $\operatorname{aff}(\mathcal{P}_i)$ denotes the affine hull of $\mathcal{P}_i$) forms a coordinate chart when its domain is appropriately restricted, due to the quantitative implicit function theorem [52]. As in [51], the preimage of each polytope serves as a conservative approximation of the domain of the chart.

Given an atlas of $\mathcal{Q}$, we can produce an atlas of $\overline{\mathcal{M}}$ by taking polytopic subsets of $\mathcal{P}_i$, whose image is a collision-free set. The IRIS algorithm naturally extends to growing polytopic regions through a general nonlinear coordinate chart [53]. The nonlinear continuation approaches used to generate the atlas of $\mathcal{Q}$ do not enforce g-convexity of the chart domains, but compactness of $\overline{\mathcal{M}}$ will guarantee the convexity radius is finite, so we can simply partition any oversized charts into g-convex pieces. We label these new polytopes $\mathcal{P}_v$, and obtain a finite atlas $\mathcal{A} = \{(\mathcal{Y}_v, \psi_v)\}_{v\in V}$ of $\overline{\mathcal{M}}$ satisfying Assumption 1. For convenience of notation, we define the inverse $\Psi_v = \psi_v^{-1}$, restricted to the appropriate domain.

We cannot satisfy Assumption 2, as $\mathcal{Q}$ may not be flat, and the chart mappings ψ_v are not isometries. However, our sets $\mathcal{P}_v$ are convex in Euclidean space, and we can replace each transition map $\tau_{u,v}$ with the affine mapping from one $\mathcal{P}_u$ to its neighbor $\mathcal{P}_v$. Any point x on the boundary between two polytopes $\mathcal{P}_u$ and $\mathcal{P}_v$ will be lifted to two different representatives y_u and y_v on the manifold by their corresponding charts. We ensure paths are well-defined as they cross the boundary between two polytopes by requiring that

$$\Psi_u(\mathcal{P}_u \cap \mathcal{P}_v) \subseteq \operatorname{dom} \psi_v. \tag{2.17}$$

We will leverage this fact in the procedure for lifting a path x_π in the PL approximation to a path y_π on $\overline{\mathcal{M}}$. So if we mirror the transformation procedure described in Subsection 2.6.1, we obtain a valid GCS problem.

Solving the shortest path problem on the PL approximation yields a path x_π as in (2.14). However, we generally do not have $\Psi_u(x_{u,1}) = \Psi_v(x_{v,0})$, so the process for lifting to a path y_π

on $\overline{\mathcal{M}}$ is more complex. We define $y_{i,0} = \Psi_i(x_{i,0})$ and $y_{i,1} = \Psi_i(x_{i,1})$, and trace the minimizing geodesic from $y_{0,0}$ towards $y_{0,1}$. At the transition from chart i to $i+1$, if $y_{i,1} \in \Psi_{i+1}(\mathcal{P}_{i+1})$, and if

$$||\psi_{i+1}(y_{i,1}) - x_{i+1,1}|| \leq ||x_{i+1,0} - x_{i+1,1}|| \,, \tag{2.18}$$

then we trace the minimizing geodesic from $y_{i,1}$ to $y_{i+1,1}$. Otherwise, we simply trace the minimizing geodesic from $y_{i,1}$ to $y_{i+1,0}$ and then to $y_{i+1,1}$.

We now endeavor to prove this GCS problem provides a useful approximation of Problem (2.7). We introduce the following notation to describe the length of a path along the PL approximation:

$$L_{\text{PL}}(x_\pi) = \sum_{i=0}^{K} ||x_{i,0} - x_{i,1}|| \,. \tag{2.19}$$

The PL approximation then becomes a metric space with distance function d_{PL} by taking the infimum of the arc lengths, analogous to (2.3). Let $p \in \mathcal{P}_p$ and $q \in \mathcal{P}_q$. Solving our approximation to global optimality yields a path x_π^* whose length $L_{\text{PL}}(x_\pi^*)$ is minimal among all paths x_π from p to q. In practice, we use the relax-and-round strategy, so we do not assume that $L_{\text{PL}}(x_\pi^*) = \mathsf{d}_{\text{PL}}(p,q)$.

We will provide an upper bound on $\mathsf{d}(\Psi_p(p), \Psi_q(q))$ in terms of $L_{\text{PL}}(x_\pi^*)$, K (x_π^* traverses $K+1$ sets per (2.14)), and the following numerical quantities derived from the PL approximation:

- ϵ_H: the maximum chart Hausdorff distance $\max_i \mathsf{d}_H(\mathcal{P}_i, \Psi_i(\mathcal{P}_i))$.
- $\alpha_{\max}$: the largest principal angle [54] between any pair of adjacent charts.

Hausdorff distance is a common metric to quantify the difference between two approximations of a geometric object [55]. The largest principal angle between charts, together with the radius of the smallest chart, presents a coarse bound on the curvature of the manifold. Taking a finer discretization generally leads to both quantities decreasing; for example, see Fig. 2.5. Furthermore, compactness of the manifold ensures that the curvature is bounded, so ϵ_H and $\alpha_{\max}$ can be made arbitrarily small with finitely many charts.

We leverage two lemmata in deriving a global upper bound on the path length (see proofs in Appendix A.3).

Lemma 1. *Fix a chart* $(\mathcal{Y}, \psi)$, *and points* $x_0, x_1 \in \psi_i(\mathcal{Y}_i)$. *Let* $y_0 = \Psi(x_0)$ *and* $y_1 = \Psi(x_1)$. *Then*

$$\mathsf{d}(y_0, y_1) \leq ||x_0 - x_1|| \left(1 + \max_{x \in \psi(\mathcal{Y})} ||D\Psi(x) - I||_{\text{op}}\right). \tag{2.20}$$

This demonstrates that the error within a chart is upper bounded by how much the total derivative of the chart mapping deviates from the identity mapping. When constructing charts as in [51], the polytope is constructed so as to lie along the tangent space of some point on the manifold. Thus, there is always some $x \in \psi(\mathcal{Y})$ such that $D\Psi(x) = I$. By smoothness, this entails that

$$\max_{x \in \psi(\mathcal{Y})} ||D\Psi(x) - I||_{\text{op}} \tag{2.21}$$

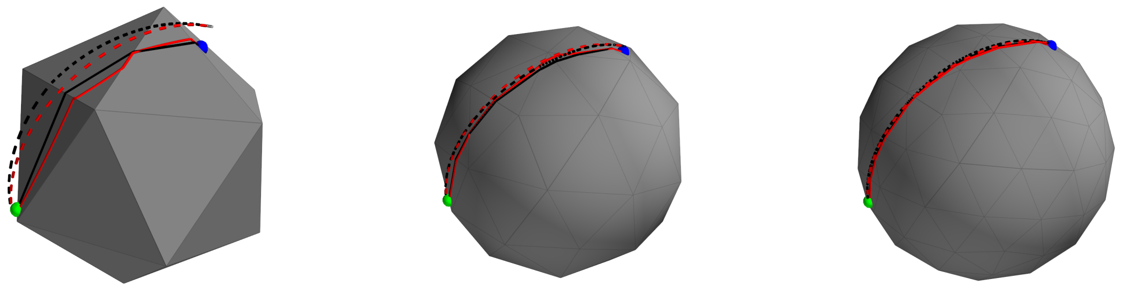


Figure 2.5: A shortest path problem on the unit sphere, solved with GgcsTrajopt using an increasingly fine discretization. The blue and green markers denote the start and end configurations, respectively. The solid black path denotes the shortest path along the mesh (found with GgcsTrajOpt), and the dotted black path is the result obtained when that path is lifted back to the sphere. Conversely, the red dotted path is the ground truth shortest path along the sphere, and the solid red path is its projection onto the PL approximation.

can be made arbitrary small by using a finer atlas of charts $(\mathcal{Y}_v, \psi_v)$ whose images have smaller diameter.

Still, an upper bound on the whole path across multiple charts can also incur error along the chart transitions. This is because we have replaced the transition maps with approximations, so the portions of the path on the manifold that cross the boundary between charts may be collapsed to zero length. The following lemma quantifies this error by projecting that portion of the path onto one of the charts.

Lemma 2. *For subsequent charts i, $i+1$ traversed by x_π,*

$$\mathsf{d}(y_{i,1}, y_{i+1,0}) \leq \epsilon_H \sin\alpha_{\max}\Big(1 + \max_{x\in\mathcal{P}_i} ||D\Psi_i(x) - I||_{\mathrm{op}}\Big). \tag{2.22}$$

We now have an upper bound on the error within a chart and across subsequent charts. Thus, by solving the PL-approximation, we obtain an optimal path length $L_{\mathrm{PL}}(x_\pi^*)$, and use this to compute an upper bound on the true distance along the manifold $\mathsf{d}(\Psi_p(p), \Psi_q(q))$.

Theorem 5. ***(Approximation Upper Bound)***

$$\mathsf{d}(\Psi_p(p), \Psi_q(q)) \leq (L_{\mathrm{PL}}(x_\pi^*) + K\epsilon_H \sin\alpha_{\max})\Big(1 + \max_{\substack{v\in V\\ x\in\mathcal{P}_v}} ||D\Psi_v(x) - I||_{\mathrm{op}}\Big), \tag{2.23}$$

where $K+1$ is the number of charts traversed by x_π^*.

Proof. Let x_π^* be the optimal path. Then we leverage the triangle inequality for the manifold distance metric, along with Lemmata 1 and 2, to obtain the desired bound:

$$\mathsf{d}(p,q) \le \sum_{i=0}^{K} \mathsf{d}(y^*_{i,0}, y^*_{i,1}) + \sum_{i=0}^{K-1} \mathsf{d}(y^*_{i,1}, y^*_{i+1,0}) \tag{2.24}$$

$$\le \sum_{i=0}^{K} \left|\left|x^*_{i,0} - x^*_{i,1}\right|\right| \left(1 + \max_{x\in\mathcal{P}_i} \left|\left|D\Psi_i(x) - I\right|\right|_{\text{op}}\right) \tag{2.25}$$

$$+ \sum_{i=0}^{K-1} \epsilon_H \sin\alpha_{\max} \left(1 + \max_{x\in\mathcal{P}_i} \left|\left|D\Psi_i(x) - I\right|\right|_{\text{op}}\right) \tag{2.26}$$

$$\le \left(L_{\text{PL}}(x^*_\pi) + K\epsilon_H \sin\alpha_{\max}\right)\left(1 + \max_{\substack{v\in V\\ x\in\mathcal{P}_v}} \left|\left|D\Psi_i(x) - I\right|\right|_{\text{op}}\right). \tag{2.27}$$

□

Roughly speaking, $\epsilon_H \sin\alpha_{\max}$ is a linear approximation of the portion of the path not accounted for within each polytope. If this quantity can be bounded above a multiplicative factor of the portion of the path in each polytope, we can obtain a multiplicative upper bound on $\mathsf{d}(\Psi_p(p), \Psi_q(q))$.

Corollary 1. *If* $\exists c > 0$ *such that* $\epsilon_H \sin\alpha_{\max} \le c \cdot \mathsf{d}(y_{i,0}, y_{i,1})$ *for all* i*, then*

$$\mathsf{d}(\Psi_p(p), \Psi_q(q)) \le L_{\text{PL}}(x^*_\pi)\,(1+c)\left(1 + \max_{\substack{v\in V\\ x\in\mathcal{P}_v}} \left|\left|D\Psi_v(x) - I\right|\right|_{\text{op}}\right). \tag{2.28}$$

2.6.5 Planning over SO(3)

The Lie group SO(3) is of great interest in robotics, and thus merits special consideration. For example, the configuration space of a ball joint is a subset of SO(3), and the configuration space of a free moving object in $\mathbb{R}^3$ is $\text{SE}(3) \cong \text{SO}(3) \times \mathbb{R}^3$. Under the standard Riemannian metric, the manifold has positive curvature (a direct result of Corollary 3.19 of [56]), and unfortunately, even a single point of positive curvature implies that the Riemannian distance function is not g-convex, even on arbitrarily small neighborhoods of that point (see proof in Appendix A.4).

Theorem 6. *Let* $\mathcal{M}$ *be a Riemannian manifold, let* $A_1 \in \mathcal{M}$ *and* $u, v \in T_{A_1}\mathcal{M}$ *such that* $\mathcal{K}(u,v) > 0$*. Then for any neighborhood* $\mathcal{U}$ *containing* A_1*,* $\mathsf{d} : \mathcal{M}^2 \to \mathbb{R}$ *is nonconvex on* $\mathcal{U}^2$*.*

One way to get around this problem is to instead use Euler angles, where a rotation is represented as the product of three rotations, about the x, y, and z axes in a given coordinate system. As a manifold, the set of Euler angles is $\mathbb{T}^3$, and it has a natural flat Riemannian metric. This makes the Euler angles parametrization very easy to plan over with GgcsTrajOpt. However, this parametrization has degeneracies due to gimbal lock, and the flat metric distorts the true distance.

Another option is to use the approximation strategy described in Subsection 2.6.4 that can handle arbitrary manifolds, which naturally encompasses SO(3). One can describe SO(3) as the subset of $\mathbb{R}^{3\times 3}$ satisfying six nonlinear constraints, but it is more common to work with more compact descriptions. One such description is the unit quaternions, which is geometrically equivalent to the 3-sphere $\mathbb{S}^3$. The unit quaternions form a double cover of SO(3), as the unit quaternions v and $-v$ correspond to the same rotation. A naive way to handle this is to simply solve the shortest path problem to the goal and its antipode, and take the shorter path. Another representation of SO(3) is the axis-angle convention, $\mathbb{S}^2 \times \mathbb{S}^1$. Each nonzero rotation corresponds to two axis-angle representations (related by having axis vectors pointing in opposite directions, and opposite angles), except for the identity rotation, which can be represented by any axis and the zero angle. Although the axis-angle convention is less elegant topologically, $\mathbb{S}^1$ is flat, so the approximation is only needed for the manifold $\mathbb{S}^2$. We empirically compare both of these approximations to the Euler angles parametrization in Subsection 2.7.5.

2.7 Experiments

We demonstrate our GgcsTrajOpt on various robotic platforms. We present illustrative toy examples of planning for a point robot on a toroidal world and an arm in the plane with multiple continuous revolute joints. We also build plans for a KUKA iiwa arm (with the base joint modified to be continuous revolute) and a PR2 bimanual mobile manipulator, implemented in Drake [57]. We make interactive recordings of these trajectories available online at our results website. For each experiment, we explicitly state the configuration space, using $\mathcal{I}$ to refer to a general bounded interval in $\mathbb{R}$. Finally, we compute shortest paths along SO(3) with each of the three approximation strategies discussed in Subsection 2.6.5.

2.7.1 Point Robot

Consider a point robot moving about a toroidal world (configuration space $\mathbb{T}^2$, modeled as a unit square with the edges identified), with convex planar obstacles. It is easy to visualize the obstacles, g-convex sets, and graph edges, as shown in Fig. 2.6. We also show an example of an optimal trajectory produced by GgcsTrajOpt, which "wraps around" the toroidal world. This plan was computed in 0.79 seconds.

2.7.2 Planar Arm

Consider a robot arm with a fixed base, composed of five continuous revolute joints (configuration space $\mathbb{T}^5$), moving through a planar workspace with convex obstacles. (We assume the arm does not suffer from self-collisions.) We present sample plans produced by GgcsTrajOpt in Fig. 2.7, together with the swept collision volumes. These two plans were found in 5.36 and 4.63 seconds, respectively. A video of these trajectories is available at our results website.

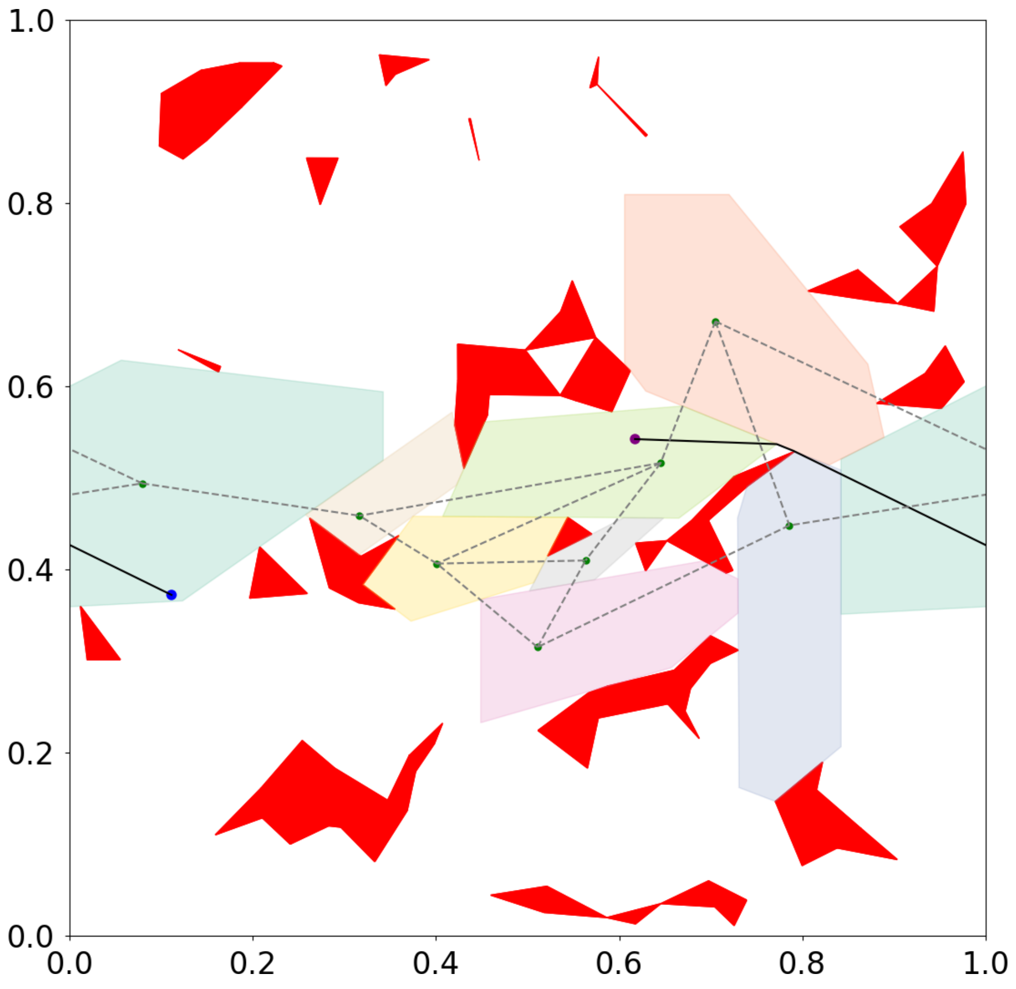


Figure 2.6: Results for a point robot in a toroidal world, realized as a unit square with opposite edges identified. Obstacles are shown in red, and each IRIS region is given a distinct pastel color. Note that one of the regions “wraps around” along the horizontal dimension, connecting opposite sides of the world. Grey dashed lines indicate which regions overlap. The optimal path between the start and end points is shown in black.

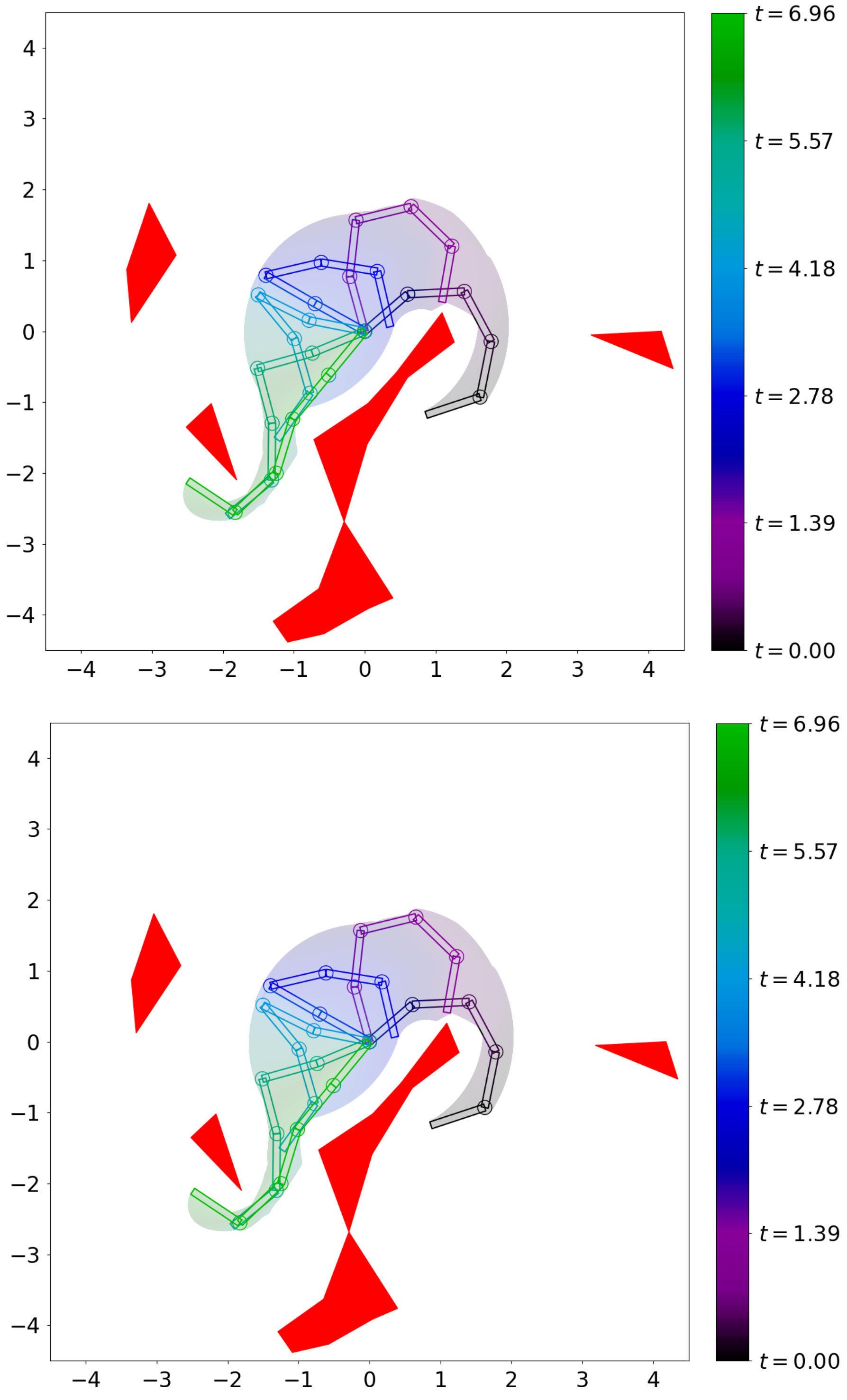


Figure 2.7: Two plans produced by GgcsTrajOpt for a planar arm around task-space obstacles (shown in red). We display both the swept collision volume and individual poses in the trajectory (colored by time, as indicated by the colorbar).

2.7.3 Modified KUKA iiwa Arm

We also demonstrate that GgcsTrajOpt can be used to plan a series of motions using a KUKA iiwa robot arm. The KUKA iiwa is a 7-DoF robot arm where each joint is a revolute joint with limits; in simulation, we remove the limits on the first joint, so the configuration space is $\mathbb{T}^1 \times \mathcal{I}^6$. We consider a scenario where the arm is mounted on a table, surrounded by three sets of shelves, with mugs arranged on the shelves. The goal of the task is to sort the mugs onto different shelves, organized by color. We specify the order of motions that are needed to achieve this goal and use GgcsTrajOpt to find the path from start to goal for each motion.

For this experiment, we used a set of 18 convex regions to achieve approximate coverage of the collision-free space. These regions were adjusted as the mugs were moved about the environment and were used to plan the complete motion of the arm – no heuristic motion or "pre-grasp pose" was needed to reach the grasp configuration. Several configurations used to seed the region generation are shown in Fig. 2.8, and the initial and final states are shown in Fig. 2.9. A video and an interactive recording of the plan are available at our results website.

The experiment consisted of 14 motions, which were each planned individually, and we use the region refinement method from [8] to account for the current placement of the mugs. This ensured that both the arm *and the grasped mug* were collision-free for the entirety of each trajectory segment. The robot takes full advantage of the base joint's lack of limits – always choosing the shortest path and never needing to unwind any rotations. For each segment, planning a trajectory took an average of 25.75 seconds (with a range of 4.63 to 50.30 seconds).

2.7.4 PR2 Bimanual Mobile Manipulator

In addition to its mobile base, the PR2 has two continuous revolute joints in each arm. We have fixed the wrist rotation and gripper joints, so the configuration space is $\mathrm{SE}(2) \times \mathbb{T}^2 \times \mathcal{I}^{10} \cong \mathbb{T}^3 \times \mathcal{I}^{12}$. We consider a scenario where the robot is driving around a square table that has an outward-pointing stack of three shelves on each side. The robot must reach into the different shelves with both arms. This represents a challenging motion planning scenario for all existing approaches due to the obstacle-rich environment and high dimensionality of the configuration space.

The performance of GgcsTrajOpt is largely driven by the choice of g-convex sets. For each set of shelves, we generate IRIS regions for the robot to reach into the top, middle, and bottom shelves with both arms simultaneously. We also generate two additional regions where the robot reaches into the middle shelf with one arm and the bottom shelf with the other while crossing its arms. Finally, we manually add various intermediate regions to promote graph connectivity and cover more of C-Free. In Fig. 2.11, we show several robot configurations along a trajectory produced by GgcsTrajOpt.

For the planning scenarios considered with the PR2, we compare GgcsTrajOpt to existing approaches. Trajectory lengths are listed in Table 2.1, plan times are listed in Table 2.2, and interactive recordings of all trajectories are available online at our results website. We compare our algorithm to kinematic trajectory optimization [13, §7.2] (abbreviated as *Drake-Trajopt*), utilizing the general nonlinear program solver SNOPT [58], [59]. Drake-Trajopt

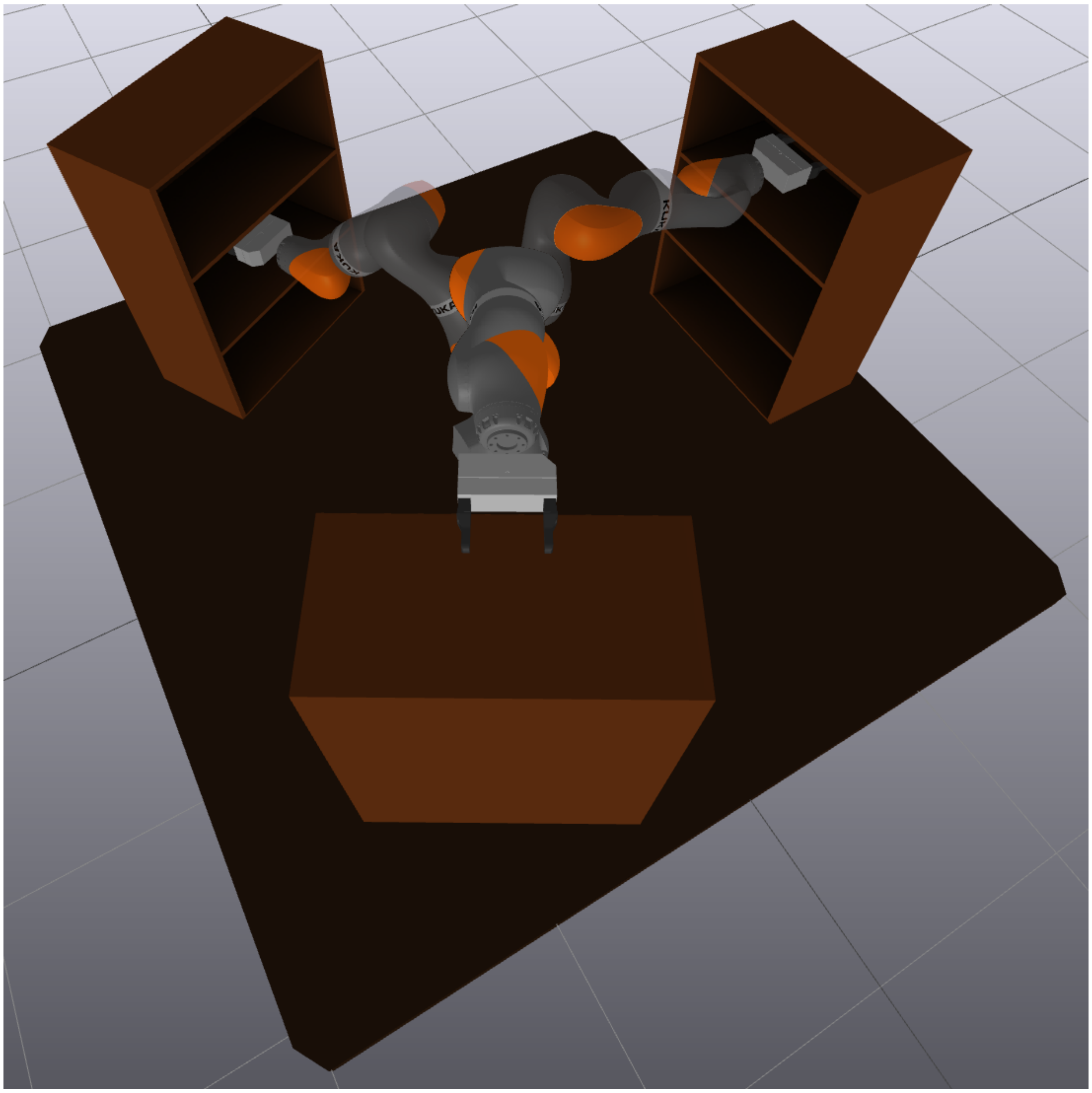


Figure 2.8: Key configurations (overlaid) used for a mug reorganization demo.

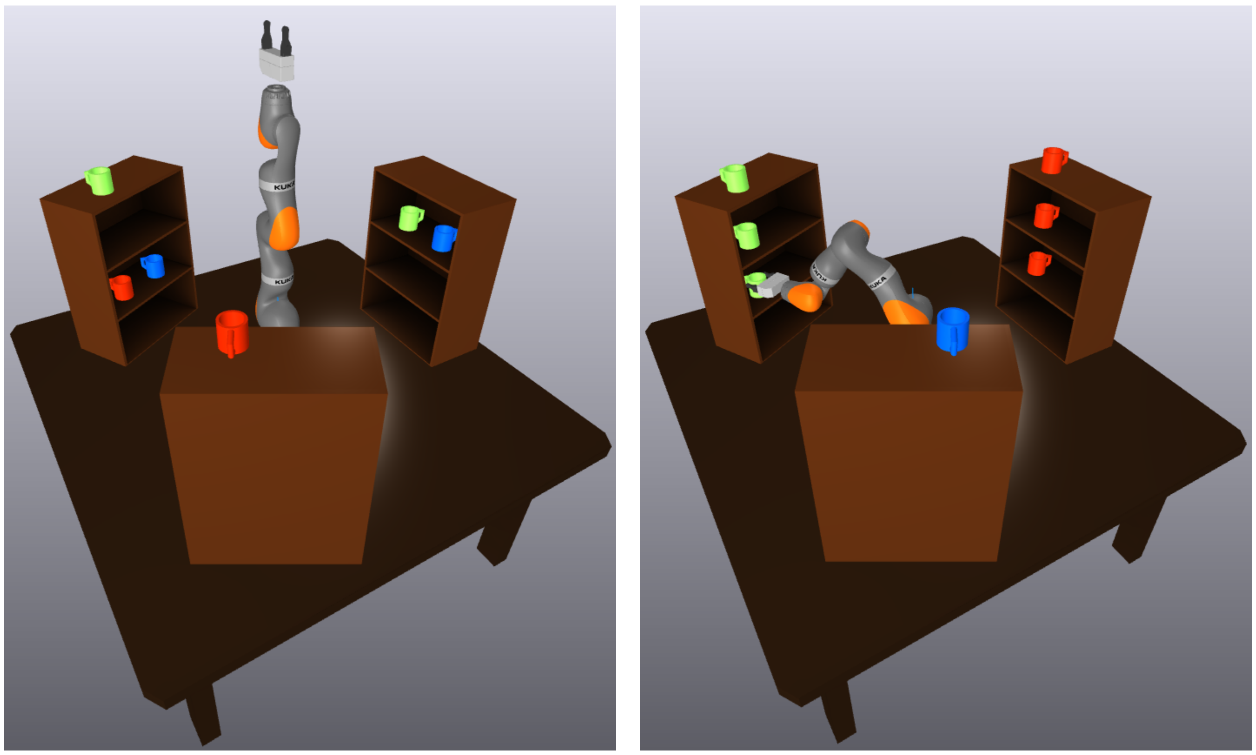


Figure 2.9: Initial (left) and final (right) states for the mug reorganization demo.

struggles to figure out how to move the arms into or out of the shelf; we often have to add waypoints to force the robot to back out of the shelf by moving its base.

We also compare it to a sampling-based PRM planner. To mitigate the curse of dimensionality and ensure connectivity between seed points, we initialize the PRM with the skeleton of the GGCS graph: for each pair of overlapping regions, we place a node in the Chebyshev center [60, page 148] of their intersection. We then add 100,000 additional samples, drawn uniformly across C-Free (with rejection sampling). This process takes 124.39 seconds. In comparison, it takes an average of 30.20 seconds to generate an IRIS region (with a range of 8.56 to 75.42 seconds). With parallelization, all of the IRIS regions were generated in only 156.63 seconds.

The plans produced by the PRM are significantly longer than those from the GgcsTrajOpt, so we also examine using the output of the PRM planner as the initial guess for the trajectory optimization. (In principle, this should help prevent Drake-Trajopt from getting stuck in local minima.) When post-processing PRM plans with Drake-Trajopt, it sometimes produces shorter final trajectories than GgcsTrajOpt, at the expense of colliding slightly with the environment (an example is shown in Fig. 2.10). This is likely due to the challenge of balancing the collision-free constraint with the minimum distance objective (and because collision-free constraints can only be applied at discrete points).

Finally, we compare our GgcsTrajOpt to two workarounds for applying ordinary GCS to non-Euclidean motion planning. One could add artificial joint limits to prevent the wraparound, but placing the joint limits incorrectly could make the optimal path infeasible. The planar arm experiment clearly demonstrates this problem; during the second trajectory

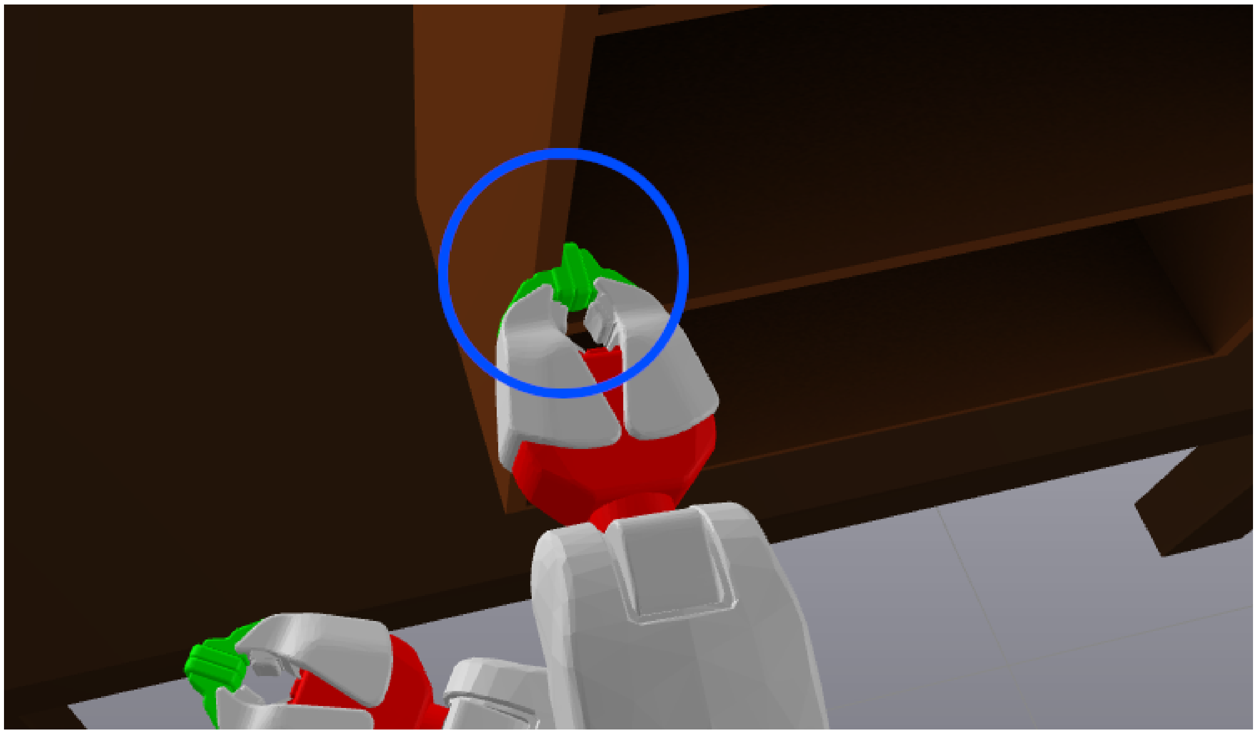


Figure 2.10: An example of the slight collisions typical of the trajectories produced by Drake-Trajopt. (The blue circle highlights the point of collision.)

in Fig. 2.7, the middle joint traverses more than 360° in the course of the plan. Thus, the optimal trajectory is infeasible for *every* possible choice of joint limits.

Another option is treating the angles as real numbers with no bound (and ignoring the fact that $0^\circ \equiv 360^\circ$). But in this case, the correct joint angle modulo 360° must be chosen to get the optimal path. Furthermore, many copies of each convex set must be made to account for each possible choice of angle modulo 360°, increasing the size of the optimization problem.

With both workarounds, a priori knowledge about the solution is required to guarantee that it is found, so in each comparison, we separately consider the best and worst cases. We use the same seed points across GgcsTrajOpt and both GcsTrajOpt workarounds.

An interesting result is that the best case for the GcsTrajOpt workarounds is sometimes slightly better than GgcsTrajOpt. This is because the sets are not bounded by the convexity radius, so they can grow larger (and cover more of C-Free) with the same seed points. If the workarounds are restricted to using the same regions as GgcsTrajOpt, then, in the best case, their performance is indistinguishable.

2.7.5 Planning over SO(3)

As discussed in Subsection 2.6.5, it is necessary to use an approximation strategy to plan over SO(3) with our methodology. To compare the efficacy of the approximation strategies,

Experiment	GgcsTrajOpt	Drake-Trajopt	PRM	PRM + Drake-Trajopt	GcsTrajOpt (Joint Limits)	GcsTrajOpt (No Joint Limits)
1T to 1B	1.829	1.803	4.359	1.808	1.826	1.839
1CL to 1CR	2.255	2.204	9.219	2.182	2.239	2.247
1M to 4M	3.875	~~6.938~~ $t = 0.275$, 5.272	14.554	~~5.874~~ $t = 0.714$, 4.381	6.482 / 10.478	3.990 / 12.589
1CL to 2CR	4.473	~~5.409~~ $t = 2.155$	12.110	~~4.108~~ $t = 0.49$	4.441 / 13.815	4.640 / 13.233
1CL to 3CR	8.182	10.263	15.250	~~7.166~~ $t = 0.7, 1.87, 2.02, 2.77$	7.820 / 12.125	8.501 / 12.125
1CL to 4CR	4.382	7.583	17.459	~~6.088~~ $t = 0.27, 0.555, 4.39$	4.728 / 9.961	4.559 / 12.418
1T to 4B	4.538	8.781	12.351	~~5.949~~ $t = 0.34, 0.68$	5.320 / 14.928	5.473 / 14.160

Table 2.1: A comparison of trajectory lengths (in configuration space) for each PR2 experiment across different methods. Experiments are titled by the start and goal configurations. The configuration names indicate the shelf positions on the table (1 through 4), followed by the position of the grippers. T, M, B, CL, and CR stand for top, middle, bottom, cross left over right, and cross right over left (respectively). Table cells that are struck through indicate that the trajectory is not collision-free, and the time stamps below the trajectory length indicate when the collisions occurred. For both GcsTrajOpt workarounds, we include the best- and worst-case results (in general, achieving the best-case results requires a priori knowledge of the optimal plan). Interactive recordings of each trajectory are available online at our results website. Each cell is linked to its corresponding recording.

Experiment	GgcsTrajOpt	Drake-Trajopt	PRM	PRM + Drake-Trajopt
1T to 1B	25.51	12.69	0.49	11.61
1CL to 1CR	39.42	15.23	0.49	16.11
1M to 4M	46.61	2.26	0.53	25.51
1CL to 2CR	62.87	9.74	0.54	21.48
1CL to 3CR	58.60	7.82	0.52	27.30
1CL to 4CR	66.15	4.32	0.54	40.10
1T to 4B	29.89	10.92	0.54	15.36

Table 2.2: A comparison of online planning times (in seconds) for each PR2 experiment across different methods. (We omit the GCS workaround comparisons, as they are indistinguishable from the corresponding GgcsTrajOpt runtimes.) Experiment names match Table 2.1.

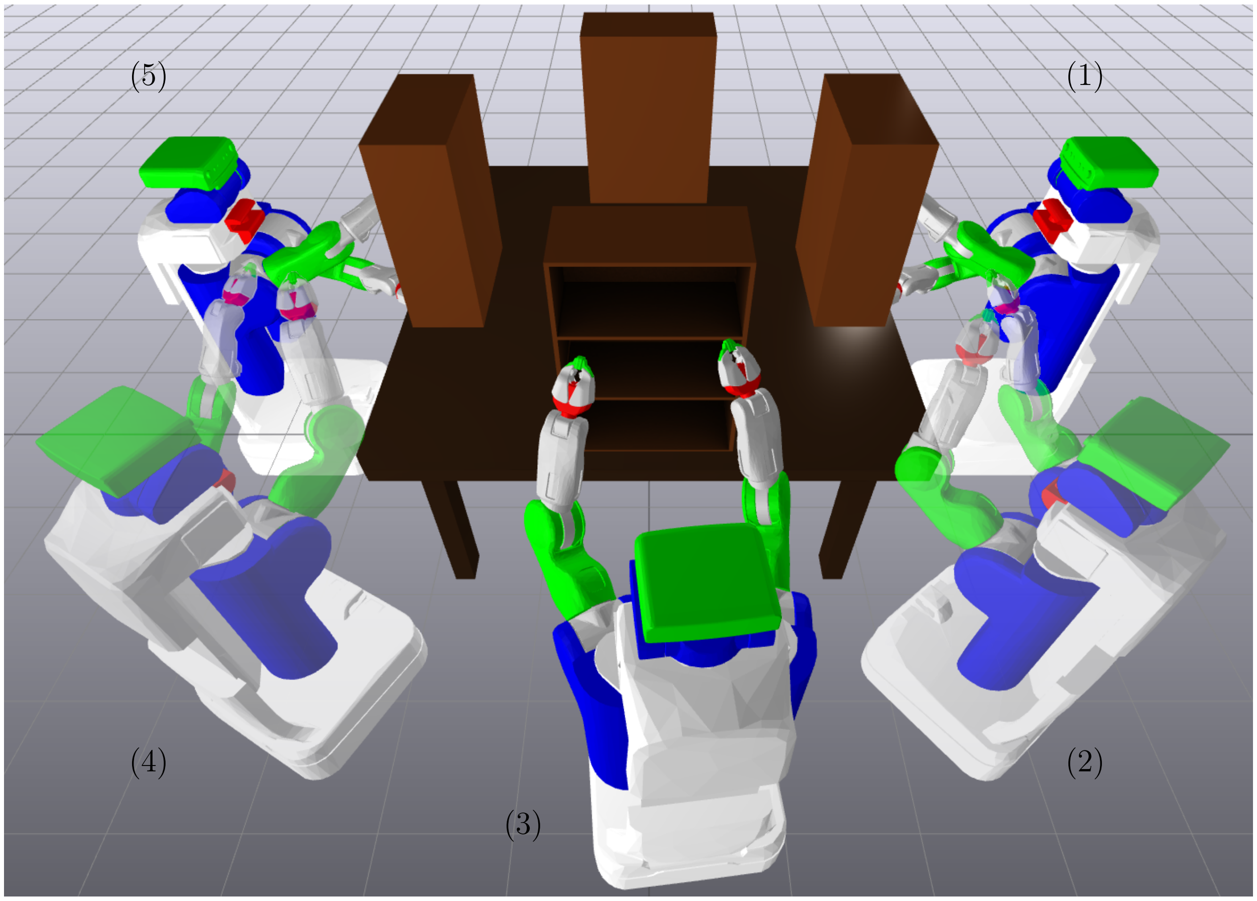


Figure 2.11: Individual poses along a trajectory produced by GgcsTrajOpt for the PR2 robot, labeled with their order in the plan.

we consider an abstract planning problem, where we have to find the shortest[1] path between two configurations in SO(3). Since there are no obstacles, we can compare the solution from each approximation to the closed-form solution, obtained from spherical linear interpolation (SLERP) [61]. For numerical purposes, we slightly expand the charts throughout these experiments, so as to achieve a positive-measure overlap.

The Euler angles description is equivalent to $\mathbb{T}^3$, which we realize as a cube with opposing faces identified. We use charts of the form $[x, x + \frac{2\pi}{3}] \times [y, y + \frac{2\pi}{3}] \times [z, z + \frac{2\pi}{3}]$ for $x, y, z \in \left\{0, \frac{2\pi}{3}, \frac{4\pi}{3}\right\}$. In our experiments, we have a GGCS with 27 sets and 702 directed edges, as the graph is fully connected.

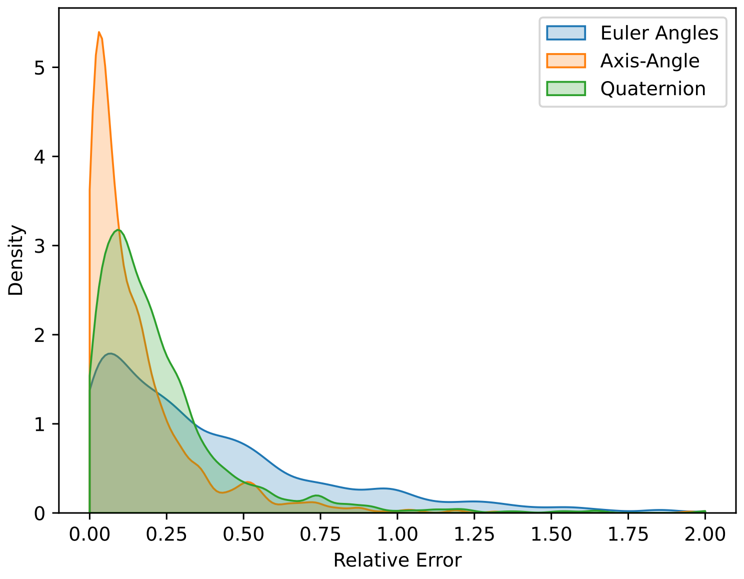
The axis-angle description is equivalent to $\mathbb{S}^2 \times \mathbb{S}^1$, where the first, 2-sphere-valued component denotes the axis of rotation and the second, circle-valued component denotes the angle rotated about that axis. We use an icosahedron as our PL approximation of the unit sphere, although higher resolution approximations can easily be constructed by subdividing the faces [62, §2.B.1]. Charts are of the form $\mathcal{P}_v \times [\theta, \theta + \frac{2\pi}{3}]$ for a face $\mathcal{P}_v$ of the PL approximation and $\theta \in \left\{0, \frac{2\pi}{3}, \frac{4\pi}{3}\right\}$. In our experiments, we have a GGCS with 60 charts and 660 directed edges. Because the axis-angle representation almost perfectly double-covers SO(3) (outside of the identity configuration), we have to solve two planning problems, to ensure we plan to the closest representative of the orientation.

The quaternion description is equivalent to $\mathbb{S}^3$. We construct a tiling with respect to the hyperspherical coordinate system [63]. We evenly tile the angular variables $\psi_1, \psi_2, \psi_3 \in [0, 2\pi]$ to a desired resolution, and then map the corners onto $\mathbb{S}^3$ (by taking the radius of each point to be 1). The polytopes $\mathcal{P}_v$ are then taken to be the convex hulls of the corners of each tile. If we subdivide each angular dimension into 3 pieces, we have 27 charts and 390 directed edges. Subdividing into 4 pieces yields 64 charts and 2240 directed edges. Similarly to axis-angle representation, $\mathbb{S}^3$ double-covers SO(3), so we have to solve two planning problems, to ensure we plan to the closest representative of the orientation.

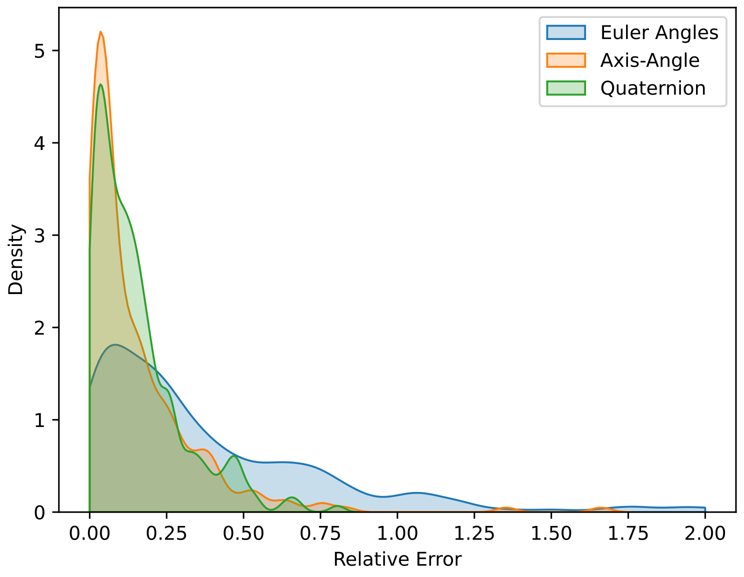
To compare these approximations, we uniformly sampled random start and goal orientations, and computed the shortest path between them for each approximation strategy. We measure their length according to the geodesic distance between each successive control point of the path, and compare to the ground truth distance between the start and goal. Ground truth for this distance metric can be obtained in closed form with spherical linear interpolation [61], allowing us to measure the approximation error for each method. In Fig. 2.12, we plot the distribution of the relative errors for each of the three methods.

The distance in the Euler angles representation is known to distort the true distance between orientations, so it is unsurprising that this choice of approximation has higher error. When the higher-resolution approximation of the quaternionic sphere is used, its relative error is roughly equivalent to that of the axis-angle approximation. However, this requires a much larger graph (and many more edges), yielding a more computationally costly optimization program. Thus, we recommend using the axis-angle parametrization, as it strikes the best balance between accuracy and computational efficiency.

[1]We measure the length of a path in SO(3) according to the canonical bi-invariant Riemannian metric.



(a) Compared across 1000 random samples, using the lower-resolution approximation of quaternionic sphere.



(b) Compared across 250 random samples, using the higher-resolution approximation of quaternionic sphere.

Figure 2.12: The distribution of relative error across many sampled start and goal configurations when planning using various SO(3) approximations strategies.

2.8 Discussion

In this chapter, we have formulated the general problem of motion planning around obstacles on Riemannian manifolds as a shortest path problem in a graph of geodesically convex sets, and we have proved this formulation inherits the same guarantees as in the ordinary Euclidean case. We describe how these theoretical developments inform simple and elegant modifications to the original GcsTrajOpt, in order to handle robots with mobile bases and continuous revolute joints. This enables us to solve motion planning problems on such robotic platforms to global optimality and guarantee that the trajectory is collision-free at every point in time. Approximate solving techniques still guarantee that trajectories are collision-free, and empirically, such trajectories are very close to optimal.

We have demonstrated that GgcsTrajOpt is a powerful tool for robot motion planning. It is capable of producing plans for high degree-of-freedom systems operating in obstacle-dense configuration spaces, such as a PR2 bimanual mobile manipulator reaching into and out of shelves. Although the planning and optimization frameworks used in GgcsTrajOpt are still in their infancy, they are already capable of producing high-quality results that are competitive with existing methods. As further research and technical improvements are made, its performance will continue to improve.

Chapter 3

Constrained Bimanual Planning with Analytic Inverse Kinematics

The work in this chapter will be published in the proceedings of the IEEE International Conference on Robotics and Automation 2024 [53].

3.1 Introduction

Enabling bimanual robots to execute coordinated actions with both arms is essential for achieving (super)human-like skill in automation and home contexts. There exists a variety of tasks that are only solvable when two arms manipulate in concert [64], such as carrying an unwieldy object, folding clothes, or assembling parts. In many manipulation tasks, one gripper can be used to provide fixture to the manipuland, while the other performs the desired action [65]; such tasks include opening a bottle, chopping vegetables, and tightening a bolt. Furthermore, some tools explicitly require two arms to use, such as hand mixers, rolling pins, and can openers.

To accomplish many of these desired tasks, the motion of the robot arms becomes subject to equality constraints imposed in task space. For example, when moving an object that is held by both hands, the robot must ensure that the transformation between the end effectors remains constant. Such task space constraints appear as complicated nonlinear equality constraints in configuration space, posing a major challenge to traditional motion planning algorithms.

In the existing literature, there are general techniques for handling task-space constraints in configuration-space planning. Sampling-based planners can project samples onto the

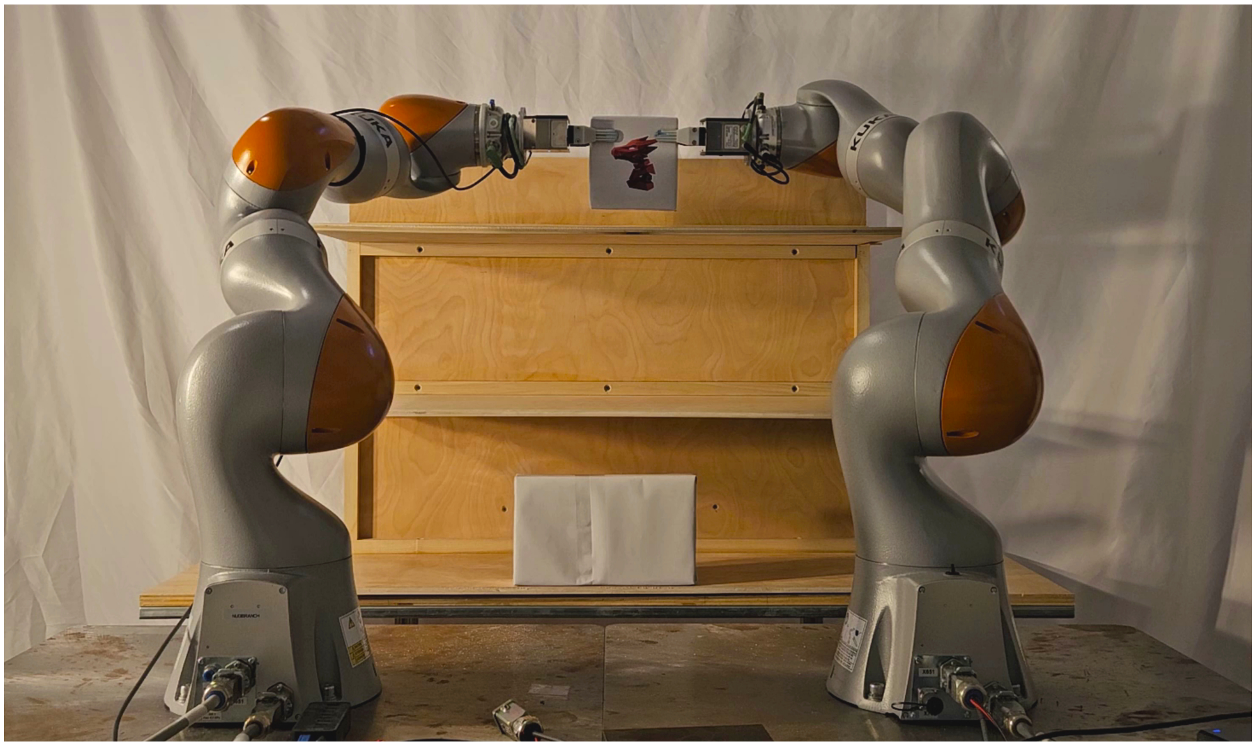


Figure 3.1: Hardware setup for our experiments. The two arms must work together to move an objects between the shelves, avoiding collisions and respecting the kinematic constraint.

constraint manifold [66] or use numerical continuation [67] to construct piecewise-linear approximations. Constraints can also be relaxed [68] or enforced directly with trajectory optimization [69]. In the case of certain bimanual planning problems, there is additional structure that is not exploited by these general methods. For certain classes of robot arms, *analytic inverse kinematics* (analytic IK) can be used to map an end-effector pose (along with additional parameters to resolve kinematic redundnacy) to joint angles in closed form. Such solutions are specific to certain classes of robot arms, but are a powerful tool to be leveraged if available. Fortunately, analytic IK *is* available for many popular robot arms available today, including the KUKA iiwa. See Fig. 3.1.

If a robot must move an object that it is holding with both hands, we propose constructing a plan for one "controllable" arm, and then the other "subordinate" arm can be made to follow it via an analytic IK mapping. Configurations where the subordinate arm cannot reach the end-effector of the primary arm, or where doing so would require violating joint limits, are treated as obstacles. In this way, we parametrize the constraint manifold so that the feasible set has positive measure in the new planning space. Because we no longer have to consider the equality constraints, sampling-based planning algorithms can be applied without modification. We can also differentiate through the IK mapping, enabling the direct application of trajectory optimization approaches.

The remainder of this chapter is organized as follows. First, we give an overview of the existing techniques used for constrained motion planning, and describe the available analytic IK solutions. Then, we present our parametrization of the constraint manifold for bimanual planning, and discuss its relevant geometric and topological properties. We describe the slight modifications which are necessary to adapt standard planning algorithms (including sampling-based planning and trajectory optimization) to operate in this framework. We then present a technique for generating convex sets in this new configuration space, such that every configuration within such a set is collision free and kinematically valid. These sets are essential for planning frameworks such as Graph of Convex Sets (GCS) [4], [9]. Finally, we present various experiments demonstrating the efficacy of these new techniques.

3.2 Related Work

Constrained bimanual planning is an instance of the general problem of motion planning with task-space constraints, a well-studied problem in robotics. There has been extensive research into sampling-based approaches; these techniques fall under a number of broad categories:

- Relax the constraints (with real or simulated compliance) to give the feasible set nonzero volume [68], [70].
- Project samples to the constraint manifold [66], [71], [72].
- Construct piecewise-linear approximations of the constraint manifold [73]–[76].
- Parametrize of the constraint manifold to eliminate constraints [77], [78].
- Build offline approximations of the constraint manifold, to simplify online planning [79], [80].

See the survey paper [81] for a more detailed summary.

Beyond sampling-based planning, standard nonconvex trajectory optimization approaches can handle arbitrary constraints, although they will generally only converge to a feasible solution with good initialization [81]. [69] used sequential convex programming on manifolds for nonconvex trajectory optimization. [82] greatly reduced constraint violations when computing trajectories on manifolds by enforcing collocation constraints in local coordinates.

Other approaches are designed specifically for the constraints that arise from robot kinematics. Inverse kinematics (IK) – computing robot joint angles so as to place the end effector at a given configuration – has been especially applicable. IK has long been used to sample constraint-satisfying configurations for bimanual robots, enabling the use of sampling-based planning algorithms [83]–[86]. IK can be leveraged to find stable, collision-free configurations for a humanoid robot [87], to help a robot arm follow a prescribed task-space trajectory [88], and to satisfy the kinematic constraints that arise when manipulating articulated objects [89]. Differential IK techniques can be used to follow task space trajectories, while satisfying constraints [90, §10.3], [91, §3.10].

A key part of our work is the use of a smooth IK mapping to parametrize the constraint manifold. Oftentimes, IK solutions are computed by solving a nonconvex mathematical program. The tools of algebraic geometry can be used to reformulate certain IK problems as systems of polynomial equations, which can be solved as eigenvalue problems [92]–[94]. However, neither of these methods yield closed-form IK solutions, nor do they guarantee smoothness. Smooth IK solutions for certain 6DoF arms can be produced by dividing the joints into two sets of three, and treating each of these as "virtual" spherical joints [95, §2.12]. IKFast [96] can be used to automatically construct analytic IK solutions for broad classes of robot arm kinematics, and is available as part of the OpenRAVE toolkit [97]. Some arms have specifically-designed geometric solutions, such as the Universal Robotics UR-5/UR-10 [98].

Robot arms with more than six degrees of freedom have kinematic redundancy – the arm can be moved while keeping its end effector fixed. This is called *self-motion* and is useful for avoiding obstacles and joint limits, but it implies that the forward kinematic mapping cannot be bijective. [99] computes a globally-consistent pseudoinverse (discarding the redundancy), but this artificially restricts the configuration space. Other approaches characterize the redundancy as a free parameter to be controlled in addition to the end-effector pose. [100] presents a strategy for treating specific joints in a 7DoF arm as free parameters, reducing the problem to that of a 6DoF arm. IKFast can discretize any additional joints. Similar to the sphere-sphere 6DoF arms, certain 7DoF arms have a sphere-revolute-sphere kinematic structure (similar to the human arm), leading to elegant geometric solutions [101], [102]. Specific geometric solutions are available for many common robot arms, including the KUKA iiwa [103], Franka Emika Panda [104], and the Barrett WAM [105].

Our parametrization can be combined with many planning algorithms to form a complete system. In this chapter, we specifically examine the canonical sampling-based planners: Rapidly-Exploring Random Trees (RRTs) [12] and Probabilistic Roadmaps (PRMs) [11]. Our contributions can also be used with the many extensions to these techniques [106]–[112]. We also describe how to use standard kinematic trajectory optimization techniques [91, §7.2], [14], [15], [113]. Finally, we describe how to extend the IRIS-NP algorithm [8] for computing convex collision-free sets to use our parametrization of the configuration space; such sets can be planned across with the GCS planning framework [4]. (These sets can also be used with other "convex set planning algorithms" [5], [114], [115].)

3.3 Methodology

We introduce a bijective mapping between joint angles and end-effector pose for a single arm with analytic IK. We then use this mapping to parametrize the set of valid configurations for constrained bimanual manipulation. The joint angles of one arm are treated as free variables for the parametrized configuration space, and the aforementioned mapping is used to determine the joint angles for the other arms (visualized in Fig. 3.2). Finally, we explain the modifications needed to adapt existing planning algorithms to utilize this parametrization.

3.3.1 Topology of Inverse Kinematics

The topological and geometric properties of inverse kinematic mappings are a classic area of study in robotics [116]–[118]. For an arm with $n \geq 6$ revolute joints, the configuration space is $\mathcal{C} \subseteq \mathbb{T}^n$, where $\mathbb{T}^n$ denotes the n-torus. The forward kinematic mapping $f : \mathcal{C} \to \mathrm{SE}(3)$ computes the end-effector pose of the arm for a given choice of each joint angle. We define the reachable set $\mathcal{X} = \{f(\theta) : \theta \in \mathcal{C}\} \subseteq \mathrm{SE}(3)$. To construct a homeomorphism between subsets

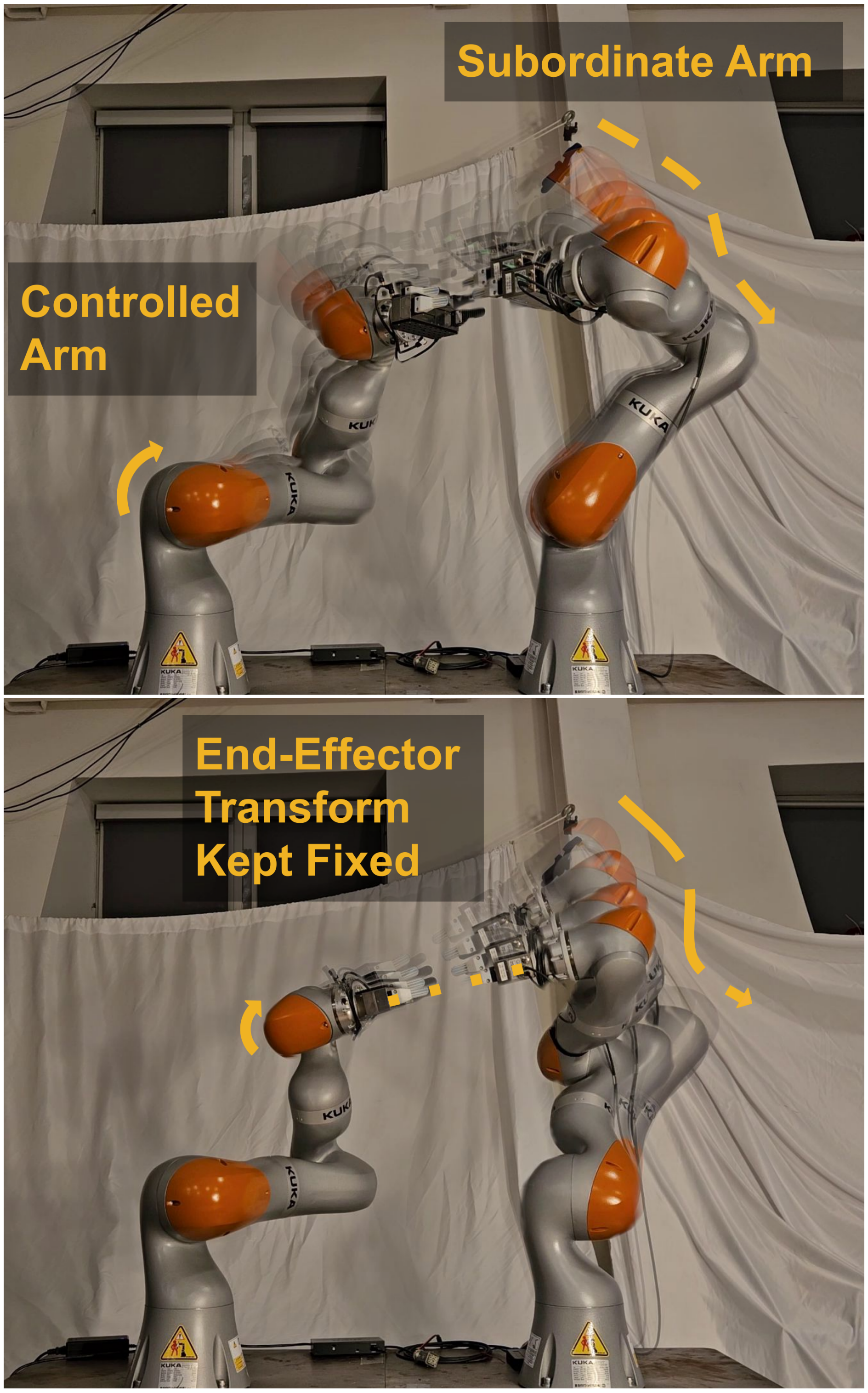


Figure 3.2: A high level description of our parametrization. The controlled arm can move freely, and analytic IK is used to position the subordinate arm so as to maintain a fixed transformation between the end-effectors.

of $\mathcal{C}$ and $\mathcal{X}$, we must restrict our domain of attention to avoid singular configurations, and augment $\mathcal{X}$ with additional degrees of freedom to match dimensions.

We give an overview of the terminology introduced in [117] for describing the global behavior of inverse kinematic mappings. A configuration for which the Jacobian of f is full-rank is called a *regular point*; otherwise, it is called a *critical point*. Because f is not injective, the preimage of a single end-effector pose may contain only critical points, only regular points, or some of both; it is respectively called a *critical value*, *regular value*, and *coregular value*. $\mathcal{W}$*-sheets* are the connected components of regular values in $\mathcal{X}$ whose boundaries are the coregular values of f. The connected components of the preimages of $\mathcal{W}$-sheets are called $\mathcal{C}$*-bundles* and are composed of regular points of $\mathcal{C}$. For a regular value $x \in \mathcal{X}$, we have

$$f^{-1}(x) = \bigcup_{i=1}^{m} \mathcal{M}_i(x), \tag{3.1}$$

where the $\mathcal{M}_i(x)$ are *self-motion manifolds* of x, so called because motion within them does not affect the end-effector pose. The label i is called the *global configuration parameter*, and a choice of $\psi \in \mathcal{M}_i(x)$ is called the *redundancy parameter*. According to [117], for robot arms in 3D space, the number of self-motion manifolds is at most 16; within a $\mathcal{C}$-bundle, the self-motion manifolds are homotopic; and if the arm has only revolute joints, then the self-motion manifolds are diffeomorphic to $\mathbb{T}^{n-6}$. (If $n = 6$, then the $\mathcal{M}_i$ are zero-dimensional, i.e., discrete points.) Examples of the continuous and discrete self motions for a 7DoF arm are shown in Fig. 3.3.

The $\mathcal{C}$-bundle/$\mathcal{W}$-sheet machinery allows us to construct well-defined IK mappings. Let $\mathcal{W}_j \subseteq \mathcal{X}$ be a $\mathcal{W}$-sheet, and let $x_0 \in \mathcal{W}_j$. Then there is an smooth injection

$$g_{i,j} : \mathcal{W}_j \times \mathcal{M}_i(x_0) \to \mathcal{C}. \tag{3.2}$$

Since the self-motion manifolds are homotopic within a $\mathcal{C}$-bundle, they are uniquely described in terms of their choice of $\mathcal{C}$-bundle and $\mathcal{W}$-sheet, so we use the shorthand $\mathcal{M}_{i,j}$ in place of $\mathcal{M}_i(x_0)$. If we let $h_{i,j}$ map joint angles to their corresponding redundancy parameter, then $(f, h_{i,j}) \circ g_{i,j}$ is the identity mapping on $\mathcal{W}_j \times \mathcal{M}_{i,j}$. Thus, with appropriate restrictions in domain and range, we have a bijection between the arm's joint angles and the product of its end-effector pose and redundancy parameters. The set $\mathcal{C}_{i,j}$, defined as the image of $g_{i,j}$, is the set of joint angles which can be handled by these mappings.

3.3.2 Parametrizing the Kinematically Constrained Space

Now, we turn our attention to the bimanual case. We use an additional subscript to denote which arm the sets and maps correspond to; for example, $\mathcal{X}_\mathbf{L}$ is the reachable set of the "left" arm, and $g_{i,j,\mathbf{R}}$ denotes the inverse kinematic mapping for the "right" arm.

When a rigid object is held with both end effectors, a rigid transformation $\mathcal{T} \in \mathrm{SE}(3)$ between them becomes fixed; we let $\phi_\mathcal{T} : \mathcal{X}_\mathbf{L} \to \mathrm{SE}(3)$ take in an end-effector pose for the left arm (henceforth called the *controlled arm*), and output the target end-effector pose for the right arm (henceforth called the *subordinate arm*). We let

$$\mathcal{X}_\mathcal{T} := \{(x, \phi_\mathcal{T}(x)) : x \in \mathcal{X}_\mathbf{L}\} \subset \mathcal{X}_\mathbf{L} \times \mathrm{SE}(3) \tag{3.3}$$

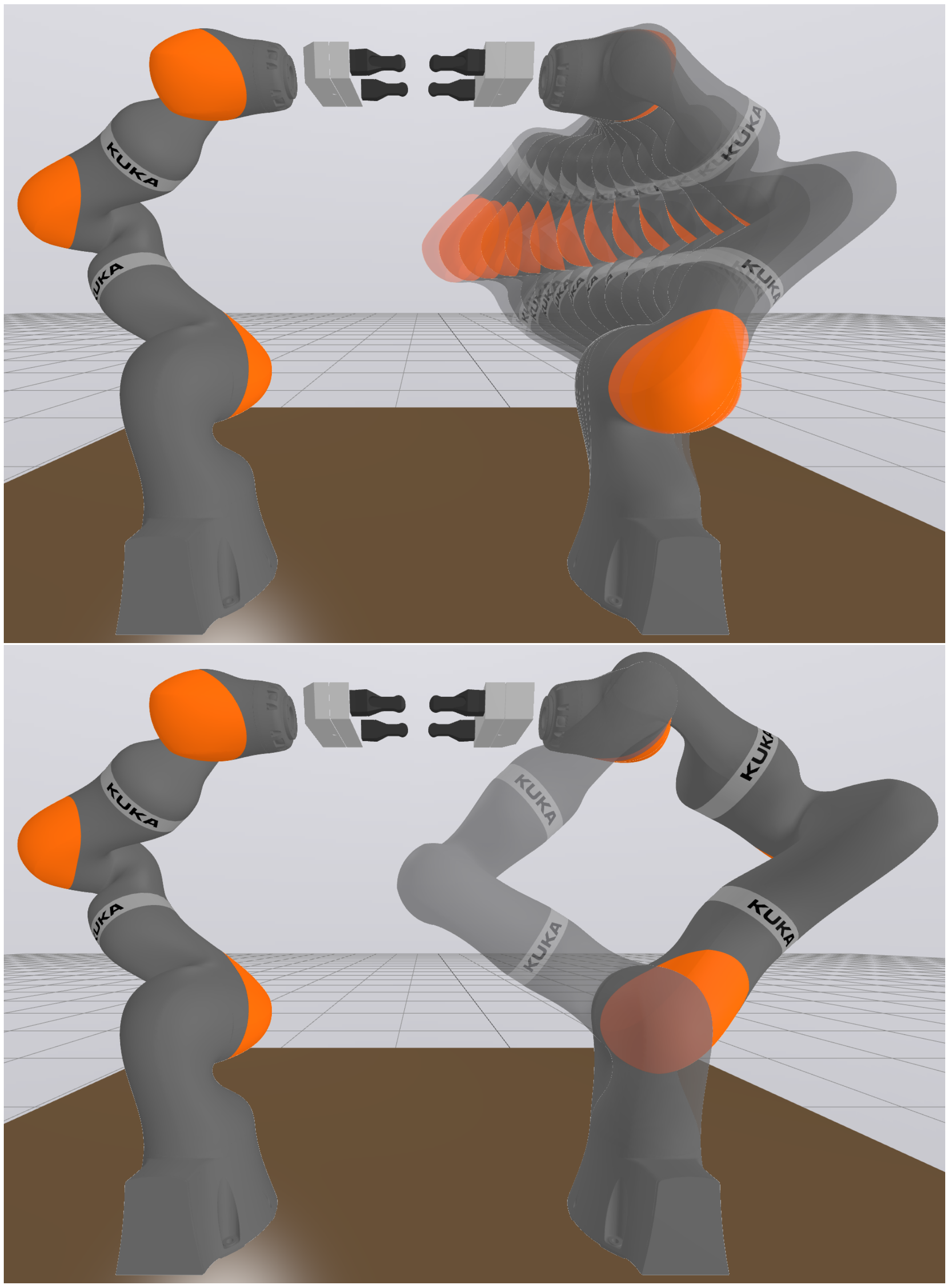


Figure 3.3: Continuous (top) and discrete (bottom) self-motions of a 7DoF arm. The continuous self-motion yields an additional degree of freedom for the planner to consider, whereas the discrete self-motion is not utilized.

denote the space of end-effector poses which are feasible for the controlled arm and for which the pose of subordinate end-effector respects transformation $\mathcal{T}$. Note that this latter pose may not be reachable for the subordinate arm, and a choice of redundancy parameter may require a violation of its joint limits. We treat both of these cases as abstract obstacles in the configuration space.

For the remainder of the chapter, we fix the global configuration parameter i and choice of $\mathcal{W}$-sheet j for the second arm. Let $\mathcal{T}$ be the desired end-effector transformation. We define a *parametrized* configuration space $\mathcal{Q} := \mathcal{C}_{\mathbf{L}} \times \mathcal{M}_{i,j,\mathbf{R}}$. $q \in \mathcal{Q}$ determines joint angles for both arms via the mapping

$$\xi : (\theta_{\mathbf{L}}, \psi_{\mathbf{R}}) \mapsto (\theta_{\mathbf{L}}, g_{i,j,\mathbf{R}}(\phi_{\mathcal{T}}(f_{\mathbf{L}}(\theta_{\mathbf{L}})), \psi_{\mathbf{R}})). \tag{3.4}$$

The individual degrees of freedom of this parametrization are shown in Fig. 3.4. For more details on why we select this specific parametrization, see Section 3.5. Let $\theta_{\min}$ and $\theta_{\max}$ be the lower and upper joint limits. A configuration $(\theta_{\mathbf{L}}, \psi_{\mathbf{R}})$ is valid if:

$$\phi_{\mathcal{T}}(f_{\mathbf{L}}(\theta_{\mathbf{L}})) \in \mathcal{W}_{j,\mathbf{R}} \quad \text{(Respect reachability.)} \tag{3.5a}$$

$$\theta_{\min} \leq \xi(\theta_{\mathbf{L}}, \psi_{\mathbf{R}}) \leq \theta_{\max} \quad \text{(Respect joint limits.)} \tag{3.5b}$$

We call the set of configurations satisfying these constraints $\mathcal{Q}_{\text{VALID}}$. For $q \in \mathcal{Q}$, if the robot is collision free for the joint angles $\xi(q)$, we say $q \in \mathcal{Q}_{\text{FREE}}$.

3.3.3 Reformulating the Motion Planning Problem

Let $\mathfrak{s}, \mathfrak{t} \in \mathcal{C}_{\mathbf{L}} \times \mathcal{C}_{\mathbf{R}}$ be the start and goal configurations. The *constrained motion planning problem* requires finding a path $\gamma = (\gamma_{\mathbf{L}}, \gamma_{\mathbf{R}}) : [0,1] \to \mathcal{C}_{\mathbf{L}} \times \mathcal{C}_{\mathbf{R}}$ by solving:

$$\begin{aligned} \operatorname{argmin} \quad & L(\gamma) && (3.6\text{a}) \\ \text{s.t.} \quad & \gamma(t) \text{ collision free} \quad \forall t \in [0,1] && (3.6\text{b}) \\ & \phi_{\mathcal{T}}(f_{\mathbf{L}}(\gamma_{\mathbf{L}}(t))) = f_{\mathbf{R}}(\gamma_{\mathbf{R}}(t)) \quad \forall t \in [0,1] && (3.6\text{c}) \\ & \gamma(0) = \mathfrak{s}, \ \gamma(1) = \mathfrak{t}. && (3.6\text{d}) \end{aligned}$$

(L denotes the arc length functional, but can be replaced with another cost.) The main challenge this formulation presents is the nonlinear equality constraint (3.6c), as this requires γ lie along a measure-zero set. Trajectory optimizers may struggle with (3.6c), and sampling-based planners must use one of the techniques described in Section 3.2.

Our *parametrized motion planning problem* is written in terms of a trajectory $\bar{\gamma} : [0,1] \to \mathcal{Q}$, with start $\bar{\mathfrak{s}}$ and goal $\bar{\mathfrak{t}}$ satisfying $\xi(\bar{\mathfrak{s}}) = \mathfrak{s}$ and $\xi(\bar{\mathfrak{t}}) = \mathfrak{t}$:

$$\begin{aligned} \operatorname{argmin} \quad & L(\xi \circ \bar{\gamma}) && (3.7\text{a}) \\ \text{s.t.} \quad & (\xi \circ \bar{\gamma})(t) \text{ collision free} \quad \forall t \in [0,1] && (3.7\text{b}) \\ & \bar{\gamma}(t) \in \mathcal{Q}_{\text{VALID}} \quad \forall t \in [0,1] && (3.7\text{c}) \\ & \bar{\gamma}(0) = \bar{\mathfrak{s}}, \ \bar{\gamma}(1) = \bar{\mathfrak{t}}. && (3.7\text{d}) \end{aligned}$$

This formulation includes the implicit requirement that the entire planned trajectory be within a single $\mathcal{C}$-bundle, due to the restricted domain of ξ. In Section 3.4, we demonstrate

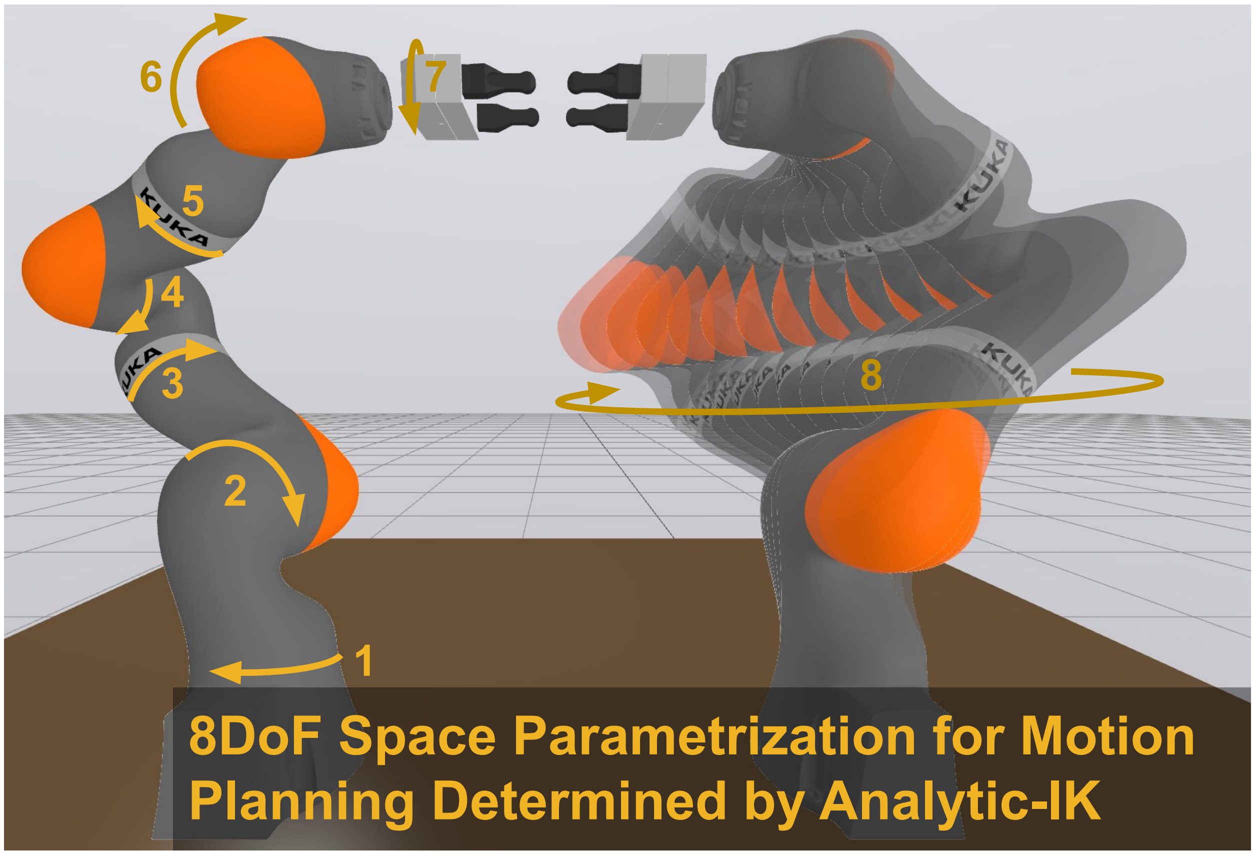

6
7
5
4
3
8
2
1
8DoF Space Parametrization for Motion Planning Determined by Analytic-IK

Figure 3.4: The individual degrees of freedom in our parametrization of the constrained bimanual configuration space.

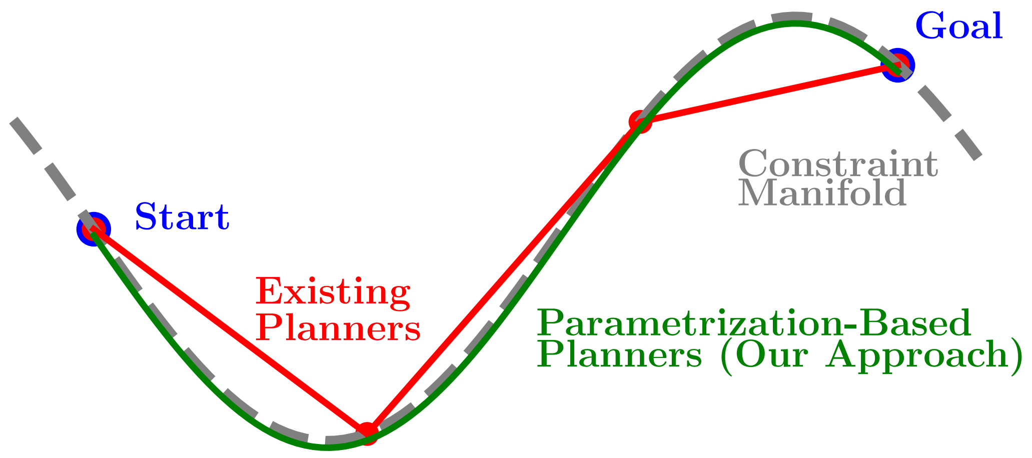


Figure 3.5: Most existing planners can only enforce constraints at discrete points along the trajectory. Parametrization-based planners (including our approach) satisfy constraints at all points by construction.

that this theoretical limitation is not a major roadblock to our framework's efficacy. A major advantage of parametrization methods is that by construction, the end-effector poses $(f_{\mathbf{L}}, f_{\mathbf{R}}) \circ \xi(\bar{\gamma}(t))$ are *guaranteed* to be related by transformation $\mathcal{T}$. For other methodologies, the constraints are only satisfied at discrete points along the trajectory. (See Fig. 3.5.)

3.3.4 Motion Planning with the Parametrization

Constraint (3.7c) is a nonlinear *inequality* constraint, so feasible trajectories are constrained to lie in a positive volume set $\mathcal{Q}_{\text{VALID}} \cap \mathcal{Q}_{\text{FREE}}$. Thus, unconstrained motion planning algorithms can function with only slight modifications.

Sampling-Based Planning

The changes required for sampling based planners can be summarized as treating points outside $\mathcal{Q}_{\text{VALID}}$ as being in collision. Because $\mathcal{Q}_{\text{VALID}} \cap \mathcal{Q}_{\text{FREE}}$ has positive measure, rejection sampling can be used to draw valid samples. When connecting samples (as in the "Extend" procedure of an RRT or "Connect" procedure of a PRM), the frequency with which collisions are checked must be adjusted, since distance in the parametrized space $\mathcal{Q}$ differs from distance in the full configuration space $\mathcal{C}_{\mathbf{L}} \times \mathcal{C}_{\mathbf{R}}$. In particular, a small motion in $\mathcal{Q}$ can lead to a relatively large motion in $\mathcal{C}_{\mathbf{L}} \times \mathcal{C}_{\mathbf{R}}$, so collision checking must be done more frequently (or at a varying scale).

Trajectory Optimization

Trajectory optimization in configuration space is already nonconvex, so implementing constraints (3.7b) and (3.7c) requires no algorithmic changes. As with sampling-based planning, collision avoidance (and other constraints applied to the full configuration space) must be enforced at a finer resolution.

Graph of Convex Sets

Let $\mathcal{U} \subseteq \mathcal{Q}_{\text{VALID}} \cap \mathcal{Q}_{\text{FREE}}$ be convex. Then the kinematic validity (and collision-free nature) of a linear path through $\mathcal{U}$ is guaranteed if its endpoints are contained in $\mathcal{U}$. Thus, the Graph of Convex Sets Planner (GCS) can function as expected with two small modifications. We minimize the arc length in the parametrized space $L(\bar{\gamma})$, as this objective provides a useful convex surrogate for the true (nonconvex) objective (3.7a). Also, for robot arms composed of revolute joints, the self-motion parameters are angle-valued, so one can either make cuts to the configuration space and treat it as Euclidean, or use the extension *Graphs of Geodesically-Convex Sets* (GGCS) [10]. The product of the angle-valued self-motion parameters will be a circle or n-torus, both of which admit a flat metric [119, p.345]. If we plan across geodesically convex (g-convex) subsets of $\mathcal{Q}_{\text{VALID}} \cap \mathcal{Q}_{\text{FREE}}$, then the problem satisfies the conditions presented in Assumptions 1 and 2 of [10]. These assumptions guarantee that the resulting path will be kinematically valid and collision-free at all times.

3.3.5 Constructing Convex Valid Sets

To use (G)GCS, one must construct (g-)convex subsets of $\mathcal{Q}_{\text{VALID}} \cap \mathcal{Q}_{\text{FREE}}$. The IRIS-NP algorithm [8] uses a counterexample search to find configurations in collision and constructs hyperplanes to separate the set from such configurations. IRIS-NP can support inequality constraints beyond collision-avoidance, but they must be inequalities. By running IRIS-NP through our parametrization, we avoid the equality constraints that would otherwise be present in our constrained bimanual manipulation problem. Given a hyperellipsoid

$$\mathcal{E}(C, d) = \{q : ||q - d||_C^2 \leq 1\}, \tag{3.8}$$

defined using the notation

$$||q - d||_C^2 = (q - d)^T C^T C (q - d), \tag{3.9}$$

a halfspace intersection

$$\mathcal{H}(A, b) = \{q : Aq \leq b\}, \tag{3.10}$$

and a *constraint set* CS, the *generalized counterexample search program* is

$$\begin{aligned} \min_q \quad & ||q - d||_C^2 && (3.11\text{a}) \\ \text{s.t.} \quad & Aq \leq b && (3.11\text{b}) \\ & q \notin \mathsf{CS}. && (3.11\text{c}) \end{aligned}$$

Given a bounding box $\mathcal{H}_0(A_0, b_0)$, a hyperellipsoid $\mathcal{E}(C, d)$ with $d \in \mathcal{H}_0(A_0, b_0)$, and a list of configuration-space constraints $\mathsf{CS}_1, \ldots, \mathsf{CS}_k$ to enforce, Algorithm 1 produces a halfspace intersection $\mathcal{H}(A, b) \subseteq \mathcal{H}_0(A_0, b_0)$ such that every point in $\mathcal{H}(A, b)$ satisfies the constraints.

We now describe the constraint sets CS needed for Algorithm 1 to generate g-convex sets in $\mathcal{Q}_{\text{VALID}} \cap \mathcal{Q}_{\text{FREE}}$, and how to encode (3.11c). For $q = (\theta_{\mathbf{L}}, \psi_{\mathbf{R}})$ (or $q = (\theta_{\mathbf{L}}, \psi_{\mathbf{R}}, \mathcal{T})$ if the

```
Input: Bounding Box $\mathcal{H}_0(A_0, b_0)$
       Hyperellipsoid $\mathcal{E}(C, d)$ s.t. $d \in \mathcal{H}_0(A_0, b_0)$
       Constraint Sets $\mathsf{CS}_1, \dots, \mathsf{CS}_k$
Output: Halfspace Intersection $\mathcal{H}(A, b)$
1 $A \leftarrow A_0$, $b \leftarrow b_0$
2 for $\mathsf{CS} = \mathsf{CS}_1, \dots, \mathsf{CS}_k$ do
3     repeat
4         $(a^\star, b^\star) \leftarrow \textsc{Solve}[(3.11), \{A, b, C, d, \mathsf{CS}\}]$
5         $A \leftarrow \textsc{VStack}(A, a^\star), b \leftarrow \textsc{VStack}(b, b^\star)$
6     until Infeasible
7 return $\mathcal{H}(A, b)$
```

Algorithm 1: Constrained IRIS (Single Iteration)

end effector transformation is allowed to vary), consider the auxiliary variable $\theta_{\mathbf{R}}$ denoting the joint angles of the subordinate arm, computed with $(\theta_{\mathbf{L}}, \theta_{\mathbf{R}}) = \xi(q)$.

First, we require that any inverse trigonometric functions used in the analytic IK mapping $g_{i,j,\mathbf{R}}$ do not violate their domains. Although this constraint would be enforced by the later constraints, specifically handling this case first greatly improves the performance of the later counterexample searches. For example, [103, Eq. 4] takes the arccos of an argument w, so we encode (3.11c) as $|w| \geq 1 + \epsilon$. When using the analytic IK solution for the KUKA iiwa, we enforce this constraint for equations (4), (6), (18), and (23) of [103].

Next, we check the joint limits (3.5b), encoded for (3.11c) as

$$\max(\xi(q) - \theta_{\max}, \theta_{\min} - \xi(q)) \geq \epsilon. \tag{3.12}$$

Finally, a configuration q is said to be *reachable* if $\phi_{\mathcal{T}}(f_{\mathbf{L}}(\theta_{\mathbf{L}})) = f_{\mathbf{R}}(\theta_{\mathbf{R}})$. Although this is an equality constraint, the set of configurations satisfying the constraint has positive volume in the parametrized space, so Algorithm 1 can still be used to generate a convex inner-approximation. For reachability counterexamples (3.5a), we compute the squared Frobenius norm of the difference between desired and realized end-effector pose, encoding (3.11c) as

$$||\phi_{\mathcal{T}}(f_{\mathbf{L}}(\theta_{\mathbf{L}})) - f_{\mathbf{R}}(\theta_{\mathbf{R}})||_F^2 \geq \epsilon. \tag{3.13}$$

These three constraints will ensure $\mathcal{H}(A, b) \subseteq \mathcal{Q}_{\text{VALID}}$. To also enforce $\mathcal{H}(A, b) \subseteq \mathcal{Q}_{\text{FREE}}$, we search for configurations q such that the robot is in collision. We separately find counterexamples for each pair of collision bodies, using equation (2) of [8]. Note that this equation operates on the full configuration $(\theta_{\mathbf{L}}, \theta_{\mathbf{R}})$, as obtained from the parametrized configuration with ξ. Because (3.11) is a nonlinear program, we solve it using SNOPT [58] with random initializations until a solution is obtained or a predefined number of consecutive failures is reached (and in that case, return infeasible).

3.4 Results

We demonstrate our new constrained planning framework using a bimanual manipulation setup with two *KUKA iiwa* 7DoF arms. Interactive recordings of all trajectories are available

Method	Top to Middle	Middle to Bottom	Bottom to Top
Trajopt	4.58*	2.85*	**4.35***
Atlas-BiRRT	4.72	5.04	6.61
Atlas-PRM	5.43	5.67	6.99
IK-Trajopt	4.24*	**1.81***	8.87
IK-BiRRT	9.91	8.69	11.42
IK-PRM	4.67	8.93	9.21
IK-GCS	**2.09**	3.32	5.62

Table 3.1: Path lengths (measured in configuration space) for each method with various start and goal configurations. Paths marked with an asterisk were not collision-free.

online at https://cohnt.github.io/Bimanual-Web/. We use the analytic IK map presented in [103]. To evaluate the merits of our IK parametrization for constrained planning, we consider a task where the two arms must move an object around a set of shelves, while avoiding collisions. We test four approaches under our parametrization:

1a. *IK-BiRRT.* We use the single-query bidirectional RRT (BiRRT) algorithm [106].

2a. *IK-Trajopt* We directly solve (3.7) with kinematic trajectory optimization [120, §10.3], using the Drake modeling toolbox [57]. We use the output of the BiRRT planner as the initial guess for the trajectory optimizer.

3a. *IK-PRM.* We use the multi-query PRM algorithm [11], initialized with nodes from multiple BiRRTs to ensure connectivity, as in [4, §C].

4a. *IK-GCS.* We use GCS-planner [4] with 19 regions, constructed from hand-selected seed points.

For both the BiRRT and PRM plans, we use short-cutting to post-process the paths [121]. We solve the GCS problems with Mosek [18]. We compare these parametrized planners with constrained planning baselines.

1b. *Constrained Trajectory Optimization.* We solve (3.6) with kinematic trajectory optimization, using the IK-BiRRT plan as the initial guess to compare with IK-Trajopt.

2b. *Sampling-Based Planning.* For sampling-based planners, we use the single-query Atlas-BiRRT and multi-query Atlas-PRM algorithms [76], as implemented in the Open Motion Planning Library [122]. The atlas and PRM are initialized from multiple Atlas-BiRRT runs.

We do not compare to any GCS baseline without IK, as the constraint manifold is inherently nonconvex; IK-GCS is the first proposal for extending GCS to this class of problems.

Constraint Violations: Because the baseline methods can only enforce the kinematic constraint at discrete points, the constraint violation can be significant between such points. The OMPL planners experienced a maximum constraint violation of 6.62 cm, and the trajectory optimization baseline experienced a maximum constraint violation of 3.22 cm. In

Method	Top to Middle	Middle to Bottom	Bottom to Top
Trajopt	10.37	5.36	7.25
Atlas-BiRRT	140.82	155.91	201.32
Atlas-PRM	0.69	0.86	0.96
IK-Trajopt	19.48	18.70	22.29
IK-BiRRT	49.42	52.53	54.10
IK-PRM	**0.46**	**0.64**	**0.61**
IK-GCS	3.41	2.32	3.32

Table 3.2: Online planning time (in seconds) for each method with various start and goal configurations. Atlas-BiRRT runtimes were only averaged over successful runs (not including timeouts).

comparison, our parametrization methods maintained all constraints within 0.001 cm. Plans from the trajectory optimization baseline also had slight collisions with obstacles.

Path Length & Planning Time: Across all methods, for various start and goal configurations, we compare path length in Table 3.1 and online planning time in Table 3.2. We ran both BiRRT methods 10 times for each plan, and report the average path length and planning time. We set a maximum planning time of 10 minutes for Atlas-BiRRT, and omit these from the averaging. Out of the 30 runs used for the table, Atlas-BiRRT timed out twice. IK-BiRRT never timed out; the longest plan took 81.17 seconds to compute. In Fig. 3.6, we visualize several plans produced by the various constrained planning algorithms.

Table 3.2 does not include offline compute time. The time to construct the Atlas-PRM varies greatly; with three random seeds, it took 33.25, 437.93, and 554.90 seconds. Constructing the IK-PRM took 2648.79 seconds, and constructing the IRIS regions for GCS took 18361.36 seconds (966.39 seconds per region on average). The IRIS region construction can also be parallelized, improving runtime.

Overall, the PRM methods have the shortest online runtimes. GCS is consistently faster than the other optimization approaches, as a result of the offline precomputation of IRIS regions. The other optimization approaches are sometimes able to find shorter paths than GCS, since they have fewer constraints, but can get stuck in local minima. Although the atlas methodologies may find shorter paths than their IK counterparts, this is at the cost of significantly higher runtimes and potentially large kinematic constraint violations.

Task Space Coverage of IRIS Regions. In Fig. 3.7, we superimpose the end-effector poses from many sampled bimanual configurations within individual IRIS regions. Despite the complicated nonlinear mapping, these convex sets are able to cover large swaths of task space, as shown in Fig. 3.7 (a). In Fig. 3.7 (b), we demonstrate that IRIS regions can reliably encompass the motions required to reach into and out of a shelf. And in Fig. 3.7 (c), we visualize an IRIS region that allows the grasp distance to vary. This is accomplished by treating the grasp distance as an additional degree of freedom in Algorithm 1. GCS can use such regions to plan motions for objects of different sizes; we include hardware demonstrations in our results video.

3.5 Discussion

We presented a novel parametrization of the constrained configuration space that arises in bimanual manipulation, which can be leveraged by both sampling-based planners and trajectory optimizers for more efficient planning. Our parametrization can be used to find shorter paths more quickly than existing approaches, and these paths will satisfy the kinematic constraints at all points along the trajectory. This parametrization also enables the use of planners such as GCS, which previously could not be applied to configuration spaces with nonlinear equality constraints.

Other parametrizations for the constrained configuration space are symmetric, and may seem more natural:

1. Treating the end-effector configuration and redundancy parameters for both arms as the free variables, and using analytic IK for both arms.

2. Treating the first four joints of each arm as free variables, and solving IK for the remaining six joints as a virtual 6DoF arm whose middle link is represented by the object held by both end-effectors.

But these choices present other disadvantages.

For the first option, we would have to choose global configuration parameters for both arms; in the case of the KUKA iiwa, this involves 64 choices (instead of the 8 options for our parametrization). Also, the shortest paths for the end effector may lead to very inefficient paths in joint space – our parametrization can at least minimize the joint space distance for one arm. Finally, it requires planning over SO(3), which is not possible for GCS (see [10, Thm. 5]).

For the second option, the choice of end-effector transformation $\mathcal{T}$ determines the kinematic structure of the virtual arm, so different grasps would require different analytic IK solutions. Constructing such solutions would be time-consuming, and they may not always exist.

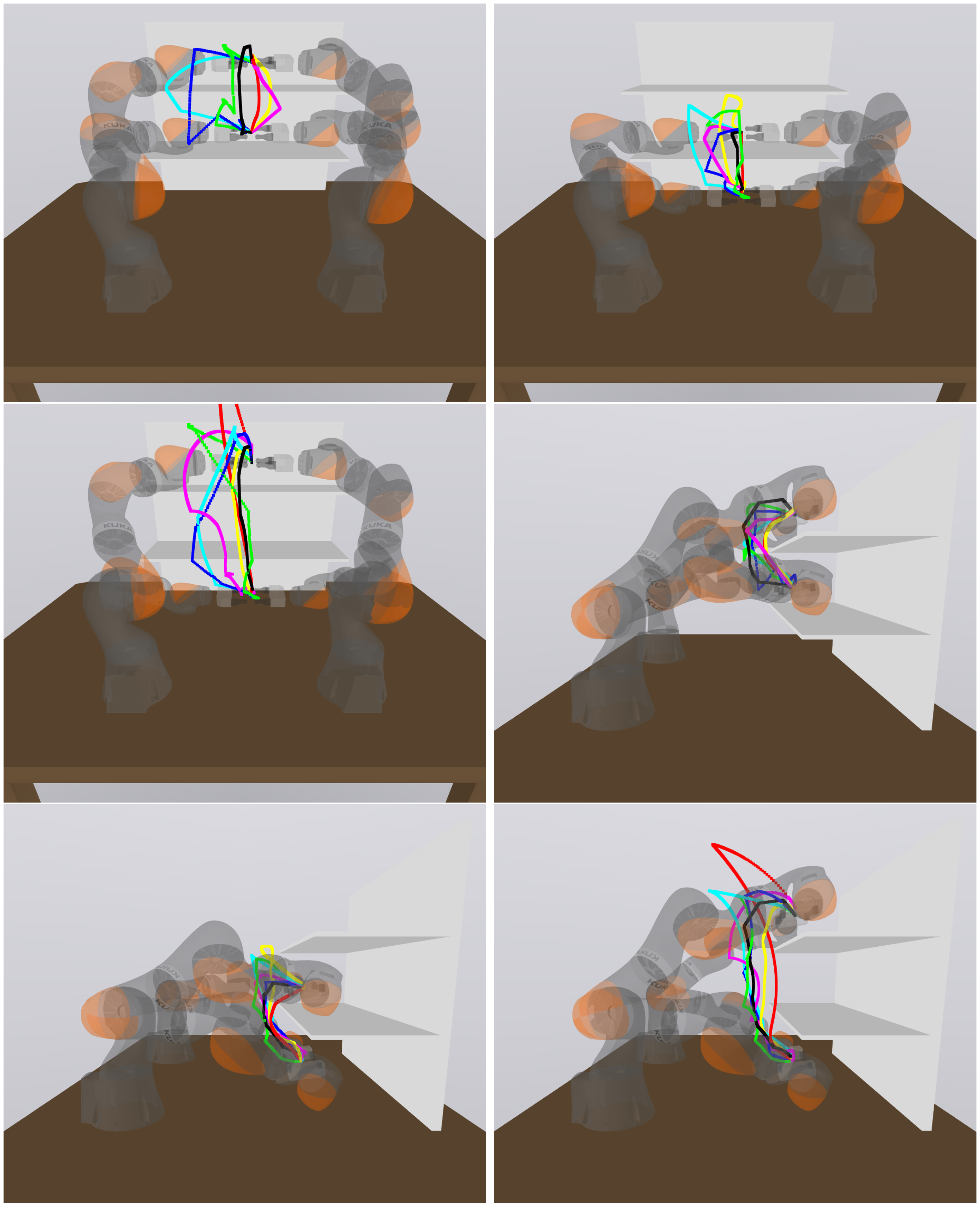
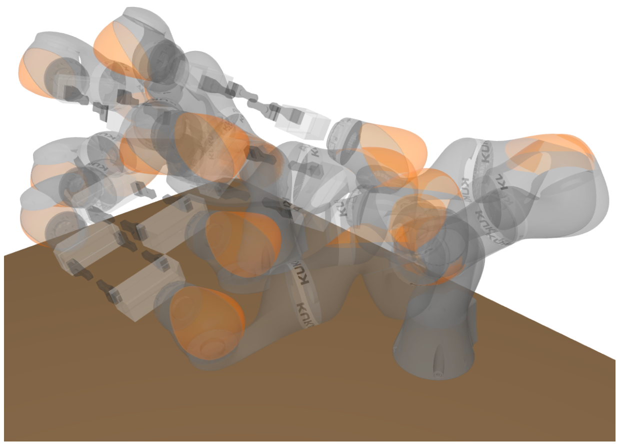
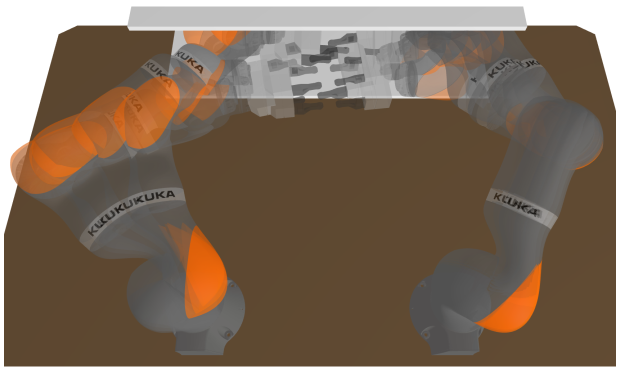


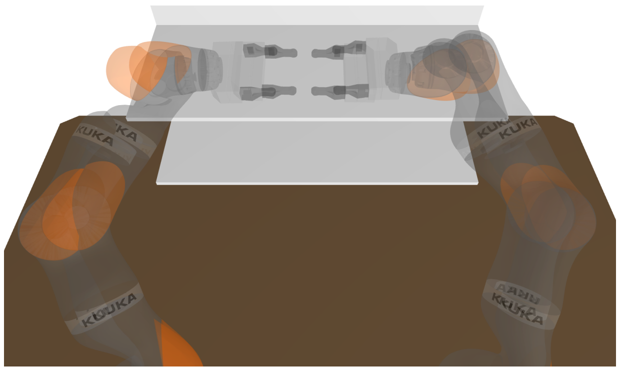
Figure 3.6: Planned trajectories for reaching into shelves. The paths denote the end effector's motion, and are colored by method. Red denotes Trajopt, green denotes Atlas-BiRRT, blue denotes Atlas-PRM, yellow denotes IK-Trajopt, pink denotes IK-BiRRT, cyan denotes IK-PRM, and black denotes IK-GCS.



(a) The coverage of a single convex region, as presented by a random collection of configurations.



(b) A collection of configurations from a region that represents motions that reach into and out of a shelf.



(c) A region that represents varying grasp distances, in addition to collision-free configurations in the shelf (not shown).

Figure 3.7: Robot configurations sampled from various IRIS regions.

Chapter 4

Conclusion

In this thesis, we have considered the problem of planning motions for robots whose configuration spaces are naturally represented as manifolds. In Chapter 1, we examined why it is necessary to generalize configuration-space planning approaches from the ordinary Euclidean case. As we summarize our contributions towards tackling this more general problem, we focus on a common theme of leveraging deep theoretical ideas to inform practical and elegant algorithmic changes, as well as a common approach in transforming a problem on a manifold into an equivalent problem in Euclidean space.

In Chapter 2, we built upon a state-of-the-art motion planning framework based on finding shortest paths through graphs of convex sets in Euclidean space. We described a natural generalization of such a framework to Riemannian manifolds and presented rigorous theoretical analysis. We proved such a representation encompasses shortest paths on Riemannian manifolds and described a reduction procedure that yields an ordinary GCS problem. We provided sufficient conditions for when such a problem is convex and can be solved to global optimality, which encompasses the configuration spaces of mobile robots and arms with continuous revolute joints. We also proved this problem is fundamentally nonconvex for a certain class of manifolds, encompassing several common robot configuration spaces (e.g. the configuration space of a ball joint or a flying robot).

In Chapter 3, we turned our attention to constrained bimanual manipulation. The configuration space that arises when an object is grasped in two hands is an implicitly-defined submanifold of the full configuration space, so most existing work has relied on numerical approximations. We presented an explicit parametrization that utilizes analytic inverse kinematics, eliminating the challenging nonlinear equality constraints. Computers can differentiate through this closed form expression, enabling optimization software to handle constraints and objectives in both the parametrized and full configuration space. As a result, the parametrization can be used with *any* motion planning algorithm.

4.1 Directions for Future Work

4.1.1 Non-Euclidean Motion Planning with Graphs of Geodesically-Convex Sets

Riemannian Optimization

As discussed in Section 2.2, we focused on constructing a optimization problem on a manifold that could easily be transformed into a convex optimization problem in Euclidean space. Another option would be to directly cast this problem as a geodesically-convex optimization problem, and leverage a Riemannian optimization framework, such as Manopt [123]. These frameworks generally assume there are no constraints besides the set constraint that decision variables lie along a given manifold. However, there has been research into geodesically-convex optimization with constraints [124], [125]. As these methods become more mature, their performance will become more competitive with the state-of-the-art (Euclidean) convex optimization frameworks. In that case, it may no longer be necessary to construct problem transformations, as these packages would be able to directly parse manifold optimization problems.

Moving Beyond Flat Manifolds

Adapting GCS to work on flat manifolds was a natural starting point. Because a flat manifold is locally isometric to a Euclidean space, there are no changes to the local geometry of the space, and the only concerns are handling global, topological differences.

Generalizing GGCS to work on non-positively curved manifolds is a clear next step. Although the underlying geometry is fundamentally different from a Euclidean space, it still has several nice properties that can allow GGCS to work. For example, the Riemannian distance function on a non-positively curved manifold is geodesically convex [126, page 4], a result which does not hold for manifolds of positive curvature. Thus, one could use the Riemannian distance function as edge weights for GGCS, in order to solve the shortest path problem. However, the equality constraints imposed in the motion planning formulation built on GGCS will no longer be affine, preventing a direct generalization.

Examples of manifolds of non-positive curvature include:

- Hyperbolic spaces, which have a constant sectional curvature of -1.
- Spaces of positive definite matrices, whose sectional curvature range from $-\frac{1}{2}$ to 0 (inclusive), with respect to the standard affine-invariant metric [127, page 71][1].

While these manifolds do not directly appear as robot configuration spaces, the GCS optimization framework has seen use for control problems [9, §8], as well other graph problems with neighborhoods [9, Appendix A]. Given the use of positive semidefinite matrices in control [128] and the use of hyperbolic spaces to represent hierarchical graph data [129], such capabilities may of both theoretical and practical interest.

[1]The lower bound is given explicitly; the upper bound follows directly from the given formula for sectional curvature, as any matrix of the form $A^T A$ is positive semidefinite, and hence has nonnegative trace.

4.1.2 Constrained Bimanual Planning with Analytic Inverse Kinematics

Broader Classes of Robot Arm Kinematics

The work in this thesis purely focused on hand-derived analytic IK solutions, constructed by examining the geometry of the arm kinematics. A more general strategy could leverage the fact that the kinematics of any robot arm can always be represented as a system of polynomial equations [7, §4.1]. This new representation can be solved via specialized computer algebra systems, such as IKFast [96]. IKFast handles kinematic redundancy by discretizing redundant joints, so finite-differencing would be necessary to differentiate through these solutions.

Alternatively, one can transform the IK problem into an eigenvalue problem, where each eigenvalue-eigenvector pair can be used to construct a solution [93, §3]. However, to come up with a consistent parametrization, it is essential to keep track of which solution is being used. Otherwise, the subordinate robot arm could "jump" between the solutions, making any resulting plan discontinuous (and thus impossible for the arms to actually follow). Fortunately, there are tools from computational algebraic geometry, such as homotopy continuation [130], that could be brought to bear on this type of problem.

Objectives Beyond Path Length

In our proposed parametrization for the constrained bimanual planning problem, the shortest path corresponds to the shortest path for the controlled arm, while the subordinate arm could take a much longer trajectory. Furthermore, the shortest path may not be the fastest path, as robots are limited in how fast they can accelerate. Optimizing for these objectives renders the problem nonconvex, potentially losing the safety and optimality guarantees we care about. However, because the constraints still describe a convex feasible set, the problem has not lost all useful structure. General nonlinear optimization techniques, such as projected gradient descent, could be used to guarantee feasibility and producing a trajectory that is locally optimal with respect to an arbitrary objective function [131]. We could then choose an objective function that more faithfully measures the distance traveled by both arms to find short paths. Instead of acceleration limits, we could use an objective function that penalizes the curvature of the path, to promote smoother and more dynamically elegant paths.

Planning Across $\mathcal{C}$-Bundles

In the constrained bimanual planning problem, the set of kinematically valid configurations inherently has singularities, and these singular configurations partition the overall configuration space into disconnected pieces [117], [118]. We have demonstrated in Section 3.4 that many motions are possible by staying within one of these pieces, and indeed, there are algorithms which intentionally avoid singularities [74]. However, we want to give the robot total freedom of motion, so we need to tackle the challenges that singularities provide to planning.

In particular, sampling-based planners struggle because the singularities form a measure-zero set. This means the probability that a random sample is drawn from one is zero, so a sampling-based planner must deliberately construct singular configurations. Also, near singularities, trajectory optimizers may suffer from numerical difficulties, as the gradients may tend to infinity. Fortunately, analytic IK algorithms and the more general eigenvalue methods introduced in Subsection 4.1.2 could be used to characterize the sets of singular configurations. The presence of discrete decisions – corresponding to which singular configurations the robot passes through – suggests that GCS could be a promising method for tackling these problems.

Appendices

A. Proofs

A.1 Proof of Theorem 1

Lemma 3. *For any $p, q \in \overline{\mathcal{M}}$, there is a piecewise-smooth path connecting p and q.*

Proof. Because $\mathcal{M}$ is path connected, there is a continuous curve $\gamma : [a, b] \to \overline{\mathcal{M}}$ joining them. Let $(U_1, \psi_1), \ldots, (U_n, \psi_n)$ be a series of charts of $\mathcal{Q}$ that cover the image of γ, with $p \in U_1$, $q \in U_n$, and $U_i \cap U_{i+1} \cap \overline{\mathcal{M}} \neq \emptyset$ for each i. (Such a finite covering exists because the image of γ is compact.) Let $t_0, \ldots, t_n \in [a, b]$ such that $t_0 = a$, $t_n = b$, and for each $i = 1, \ldots, n-1$, $\gamma(t_i) \in U_i \cap U_{i+1}$. For each $i = 1, \ldots, n-1$, let $\tilde{\gamma}_i$ be a smooth curve joining $\psi_i(\gamma(t_i))$ to $\psi_{i+1}(\gamma(t_{i+1}))$ that is contained within $\psi_i(U_i \cap \overline{\mathcal{M}})$. Let $\tilde{\gamma}_0$ join $\psi_1(\gamma(t_0))$ to $\psi_1(\gamma(t_1))$ and be contained within $\psi_1(U_1 \cap \overline{\mathcal{M}})$, and let $\tilde{\gamma}_n$ join $\psi_n(\gamma(t_{n-1}))$ to $\psi_n(q)$ and be contained within $\psi_n(U_n \cap \overline{\mathcal{M}})$. Then by lifting each of these curves to $\overline{\mathcal{M}}$, and concatenating them, we obtain a piecewise-smooth curve connecting p and q. □

Proof of Theorem 1. The proof follows by verifying that $\overline{\mathcal{M}}$ is a complete, locally compact length space, so that we can apply Theorem 2.5.23 of Burago, Burago, and Ivanov [47, page 50]. A length space is a metric space in which the distance between any two points is given by the infimum of the arc lengths of all paths connecting those two points. A length space is complete if the distance between any two points is finite. Thus, $\overline{\mathcal{M}}$ inherits a length structure from $\mathcal{Q}$ (with the restriction to curves that are entirely contained in $\overline{\mathcal{M}}$). All topological manifolds are locally compact. To check that $\overline{\mathcal{M}}$ is complete, let $p, q \in \overline{\mathcal{M}}$. By Lemma 3, there is a piecewise-smooth curve connecting p and q, so the set of arc lengths of curves connecting p and q is nonempty. It is also bounded below, so its infimum is finite, and thus $\mathsf{d}(p, q)$ exists. We conclude that $\overline{\mathcal{M}}$ is a complete, locally compact length space. □

A.2 Proof of Theorem 4

Lemma 4. *Let $(\mathcal{Y}_1, \psi_1)$ and $(\mathcal{Y}_2, \psi_2)$ be coordinate charts of $\mathcal{M}$, with ψ_1 and ψ_2 local isometries, and $\mathcal{Y}_1$ and $\mathcal{Y}_2$ g-convex. Then there is a Euclidean isometry ξ such that $\forall p \in \mathcal{Y}_1 \cap \mathcal{Y}_2$, $\psi_1(p) = (\xi \circ \psi_2)(p)$.*

Proof. $\mathcal{Y}_1 \cap \mathcal{Y}_2$ is g-convex, and hence connected. $\psi_1 \circ \psi_2^{-1}$ is a local isometry between two connected open subsets of Euclidean space (with appropriate restriction of domain and range), so $(\psi_1 \circ \psi_2^{-1})_{*,p}$ is an orthogonal matrix for any p. Thus, we can apply Theorem 1.8-1 of Ciarlet [132, page 44]. □

Lemma 5. *Consider $\mathcal{Y} \subseteq \mathcal{Z} \subseteq \mathcal{M}$, where $\mathcal{Z}$ is g-convex, and we have a coordinate chart $(\mathcal{Z}, \psi)$ such that ψ is a local isometry. If $\psi(\mathcal{Y})$ is convex, then $\mathcal{Y}$ is g-convex.*

Proof. Fix $p, q \in \mathcal{Y}$. Then there is a unique minimizing geodesic γ connecting p to q, and γ is contained in $\mathcal{Z}$. Because ψ is a local isometry, it maps γ to a line segment in $\psi(\mathcal{Z})$. $\psi(p), \psi(q) \in \psi(\mathcal{Y})$, so by convexity of $\psi(\mathcal{Y})$, $\psi \circ \gamma$ is contained in $\psi(\mathcal{Y})$. Thus, γ is contained in $\mathcal{Y}$, so $\mathcal{Y}$ is g-convex. □

Proof of Theorem 4. For each $i \in [m]$, we can construct a ball $B_{s_i}(c_i) \supseteq \mathrm{proj}_{\mathcal{Q}_i}(\mathcal{Y})$, with $s_i < r_i$. Define $\mathcal{Z} = \prod_{i\in[m]} B_{s_i}(c_i)$, a g-convex set. Consider the Riemannian normal coordinates of $\mathcal{Q}$ at $(c_1, \ldots, c_m)$. This coordinate system, restricted to $\mathcal{Z}$, induces a coordinate chart φ. Because $\mathcal{Q}$ is flat, φ is a local isometry, so by Lemma 4 there is a Euclidean isometry ξ such that $\varphi(\mathcal{Y}) = \xi(\psi(\mathcal{Y}))$, so $\varphi(\mathcal{Y})$ is convex. Thus, by Lemma 5, $\mathcal{Y}$ is g-convex. □

A.3 Proofs of Lemmata 1 and 2

Proof of Lemma 1. Define $\gamma(t) = tx_0 + (1-t)x_1$. Then $(\Psi_i \circ \gamma)(0) = y_0$ and $(\Psi \circ \gamma)(1) = y_1$, so $\mathsf{d}(y_0, y_1) \leq L(\Psi \circ \gamma)$. Let $\Delta x = x_1 - x_0$. Well,

$$\begin{aligned} ||(\Psi \circ \gamma)'(t)|| &= ||D\Psi(t)\gamma'(t)|| = ||D\Psi(t)\Delta x|| && \text{(A.1)} \\ &= ||\Delta x + (D\Psi_i(t) - I)\Delta x|| && \text{(A.2)} \\ &\leq ||\Delta x|| + ||(D\Psi(t) - I)\Delta x|| && \text{(A.3)} \\ &\leq ||\Delta x|| + ||D\Psi(t) - I||_{\mathrm{op}} \, ||\Delta x|| && \text{(A.4)} \end{aligned}$$

where we have leveraged the triangle inequality (A.3) and the operator norm inequality (A.4). By considering the largest possible operator norm over $\psi(\mathcal{Y})$, we obtain

$$\begin{aligned} L(\Psi \circ \gamma) &\leq \int_0^1 ||\Delta x|| + ||D\Psi(t) - I||_{\mathrm{op}} \, ||\Delta x|| \, \mathrm{d}t && \text{(A.5)} \\ &\leq \int_0^1 ||\Delta x|| \left(1 + \max_{x\in\psi(\mathcal{Y})} ||D\Psi(x) - I||_{\mathrm{op}}\right) \mathrm{d}t && \text{(A.6)} \\ &= ||\Delta x|| \left(1 + \max_{x\in\psi(\mathcal{Y})} ||D\Psi(x) - I||_{\mathrm{op}}\right), && \text{(A.7)} \end{aligned}$$

which is the desired upper bound. □

Proof of Lemma 2. Construct a triangle whose three vertices are $x_{i,1} = x_{i+1,0}$, $y_{i,1}$, and $\psi_{i+1}(y_{i,1})$. We note that $\angle x_{i,1} y_{i,1} \psi_{i+1}(y_{i,1})$ is equal to the angle between $\mathcal{P}_i$ and $\mathcal{P}_{i+1}$ along the plane through those three points (which we call α), and hence is bounded above by $\alpha_{\max}$.

$$||x_{i+1,0} - \psi_{i+1}(y_{i,1})|| = ||x_{i+1,0} - y_{i,1}|| \sin\alpha \leq \epsilon_H \sin\alpha_{\max} \tag{A.8}$$

Applying Lemma 1 to $x_{i+1,0}$ and $\psi_{i+1}(y_{i,1})$ completes the proof. □

A.4 Proof of Theorem 6

Proof of Theorem 6. Fix a neighborhood $\mathcal{U}$ of A_1. Since the sectional curvature is invariant with respect to a change of basis, suppose without loss of generality that $||u|| = ||v|| = 1$ and $\langle u, v\rangle = 0$. To prove this result, we will construct a geodesic γ on $\mathcal{U}^2$ such that $\mathsf{d} \circ \gamma$ achieves smaller values on its endpoints than at its center. This relies on the properties of specially constructed Levi-Civita parallelogramoids on $\mathcal{U}$.

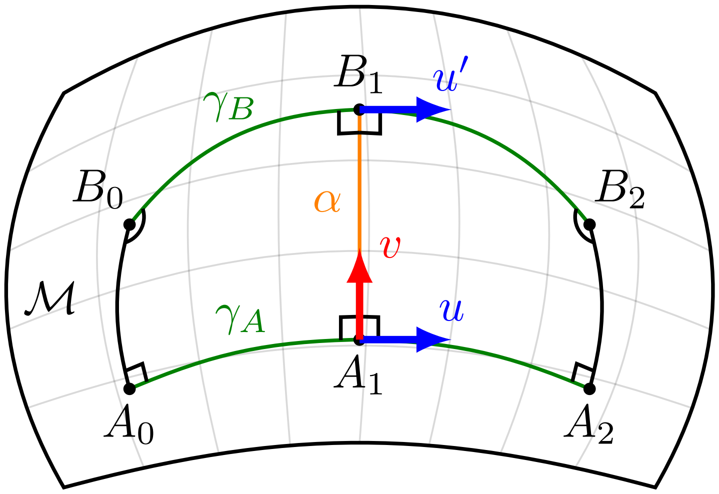


Figure A.1: The construction of two Levi-Civita parallelogramoids used in the proof of Theorem 6. $\mathsf{d}(A_0, B_0) < \mathsf{d}(A_1, B_1)$ and $\mathsf{d}(A_2, B_2) < \mathsf{d}(A_1, B_1)$, which demonstrates the nonconvexity of d around A_1. $\angle A_0B_0B_1$ and $\angle A_2B_2B_1$ are obtuse.

Let $\epsilon > 0$ be small, such that $\exp_p(\overline{B_{2\epsilon}(0)}) \subseteq \mathcal{U}$. Let $\alpha : t \mapsto \exp_p(tv)$, let $B_1 = \alpha(\epsilon)$, and let $u' = \Gamma(\alpha)_0^\epsilon(u)$ be u parallel transported from A_1 to B_1. Let $\gamma_A : t \mapsto \exp_{A_1}(tu)$ and $\gamma_B : t \mapsto \exp_{B_1}(tu')$, with domain $[-\epsilon, \epsilon]$. Then $\gamma = (\gamma_A, \gamma_B)$ is a geodesic of $\mathcal{U}^2$. Define $A_0 = \gamma_A(-\epsilon)$, $B_0 = \gamma_B(-\epsilon)$, $A_2 = \gamma_A(\epsilon)$, and $B_2 = \gamma_B(\epsilon)$. This construction is visualized in Fig. A.1. We want to show that $\mathsf{d}(A_0, B_0) < \mathsf{d}(A_1, B_1)$ and $\mathsf{d}(A_2, B_2) < \mathsf{d}(A_1, B_1)$.

The points A_1, B_1, A_2, and B_2 describe a Levi-Civita parallelogramoid, with base A_1B_1 and suprabase A_2B_2. Thus, we can relate the length of the base and suprabase via the formula of [133, page 244]

$$\mathsf{d}(A_2, B_2)^2 = \mathsf{d}(A_1, B_1)^2 + \tfrac{8}{3}\langle \mathcal{R}(\epsilon u, \epsilon v)\epsilon u, \epsilon v\rangle + O(\epsilon^5) \tag{A.9}$$

Because $||u|| = ||v|| = 1$ and $\langle u, v\rangle = 0$,

$$\langle \mathcal{R}(\epsilon u, \epsilon v)\epsilon u, \epsilon v\rangle = -\epsilon^4 \langle \mathcal{R}(u, v)v, u\rangle = -\epsilon^4 K(u, v) < 0 \tag{A.10}$$

So as ϵ is decreased towards 0, the fifth and higher order terms vanish, and $\mathsf{d}(A_2, B_2) < \mathsf{d}(A_1, B_1)$. A similar calculation shows that $\mathsf{d}(A_0, B_0) < \mathsf{d}(A_1, B_1)$. Thus, $d \circ \gamma$ has a local minimum, so we conclude that d is nonconvex on $\mathcal{U}$. $\square$

B. Experiment Implementation Details

In this appendix, we present further details about the setup of our experiments and demonstrations.

B.1 Planar Arm

The trajectories shown in Subsection 2.7.2 were generated with a GGCS that had 19 sets. We generated IRIS regions for the start and goal configurations, and hand-picked several seed points along the narrow gap between the two lower obstacles to help ensure connectivity between the start and goal. We then generated the remaining IRIS regions with random seed points (chosen uniformly from C-Free with rejection sampling).

The GgcsTrajOpt results shown used the sum of the path length and path duration as the objective. We used the relax-and-round approximation strategy to produce the trajectories shown in the results. The first trajectory had a path length of 7.749, and the second had a path length of 8.448. When solving the integer program with branch-and-bound, the first trajectory had a path length of 7.274, and the second had a path length of 8.008. (Note that the optimal solution for the latter trajectory still had the middle joint of the arm traverse more than 360°.)

B.2 KUKA iiwa Arm

The motions shown in Subsection 2.7.3 used regions generated from 18 seed points. The seeds consisted of one seed for each middle and top shelf (the bottom shelves are excluded because they are kinematically unreachable), one seed above each shelf, one seed directly between each shelf, and two seeds per shelf to aid moving between the top and middle shelves. Regions were generated for each seed with both an empty hand and a mug in the hand to aid both types of trajectory planning. Regions were post-processed to remove redundant hyperplanes with the `ReduceInequalities` algorithm from Drake.

GgcsTrajOpt minimized both time and path length of the trajectory while ensuring continuity of the path, velocity, and acceleration. For velocity limits, the real velocity limits of the KUKA iiwa hardware were used. Trajectories were computed using the relax-and-round approximation strategy.

B.3 PR2 Bimanual Mobile Manipulator

To model the PR2 robot, we use the URDF file and object meshes included with Drake. For each link, we take the convex hull of the mesh and use that as the collision geometry. (Collisions annotated in Table 2.1 are determined based on the true collision geometry, not the convex hulls.) The plans we produce take into account the robot's base joint, torso lift joint, and all arm joints (up to the final wrist rotation joint and gripper joints). All other joints are fixed.

For the experiments demonstrated in Subsection 2.7.4, we first constructed IRIS regions for each of the possible goals: reaching into each of the three shelves in a set with both arms, crossing right-over-left on the middle and bottom shelves, and crossing left-over-right. (See Fig. 2.1 for a visualization of these cross-over poses.) We then hand select a few intermediate seed points; the regions around these points are used to promote connectivity among the various shelf-reaching regions. We construct these regions for each set of shelves, except for the experiments where the start and goal are within the same set of shelves.

We take several actions to improve the efficiency of GgcsTrajOpt. To reduce the number of constraints needed, we simplify the IRIS regions by removing redundant halfspaces from their polyhedral representation, using the `ReduceInequalities` algorithm in Drake. We also only include shelf-reaching regions if they are the start or goal of the plan. This greatly reduces the size of the optimization problem, promoting faster solve times. Empirically, it also leads to a shorter trajectory, likely due to a tightening of the convex relaxation. For GgcsTrajOpt, we use the same objective as the planar arm experiments (the sum of the trajectory length and duration), and we use the relax-and-round strategy.

For the comparison to kinematic trajectory optimization (Drake-Trajopt), we use the same objective as GgcsTrajOpt: the sum of the trajectory duration and length. However, the trajectories are parametrized as B-splines instead of linear segments (or Bézier curves if the extensions in Subsection 2.6.3 are utilized). The `KinematicTrajectoryOptimization` can automatically construct the nonlinear optimization problem for a given scenario, which we then solve with SNOPT. We first solve the problem without collision-free constraints. The output of this initial problem is used as the initial guess for the full problem (i.e., including collision-free constraints). The collision-free constraint is encoded with the `MinimumDistanceConstraint` class. We set a minimum distance of 1mm and begin applying a penalty at 1cm. This constraint is applied to 50 points along the trajectory. (Such a constraint can only be evaluated pointwise.) For motion planning tasks where the robot had to move between shelves, Drake-Trajopt was unable to produce a collision-free trajectory. Thus, we added waypoints near the beginning and end of the trajectory, in which the robot was in the same configuration as the start and goal (respectively), but the base was moved away from the shelf. This was only sometimes effective at finding collision-free trajectories.

As in [4], we use the PRM planner from the Common Robotics Utilities library [134], with the modifications described in Subsection 2.7.4. Given a piecewise-linear trajectory from the PRM, we construct a B-spline that passes through the nodes on this trajectory for use as an initial guess for Drake-Trajopt. In this case, when solving the optimization problem, we begin applying a distance penalty at 1m and perform collision checking at 100 points along the trajectory.

References

[1] T. Lozano-Perez, *Spatial planning: A configuration space approach.* Springer, 1990.

[2] T. Möller, “A fast triangle-triangle intersection test,” *Journal of graphics tools*, vol. 2, no. 2, pp. 25–30, 1997.

[3] E. G. Gilbert, D. W. Johnson, and S. S. Keerthi, “A fast procedure for computing the distance between complex objects in three-dimensional space,” *IEEE Journal on Robotics and Automation*, vol. 4, no. 2, pp. 193–203, 1988.

[4] T. Marcucci, M. Petersen, D. von Wrangel, and R. Tedrake, “Motion planning around obstacles with convex optimization,” *Science robotics*, vol. 8, no. 84, eadf7843, 2023.

[5] T. Marcucci, P. Nobel, R. Tedrake, and S. Boyd, “Fast path planning through large collections of safe boxes,” *arXiv preprint arXiv:2305.01072*, 2023.

[6] R. Deits and R. Tedrake, “Computing large convex regions of obstacle-free space through semidefinite programming,” in *Algorithmic foundations of robotics XI*, Springer, 2015, pp. 109–124.

[7] A. Amice, H. Dai, P. Werner, A. Zhang, and R. Tedrake, “Finding and Optimizing Certified, Collision-Free Regions in Configuration Space for Robot Manipulators,” in *International Workshop on the Algorithmic Foundations of Robotics*, Springer, 2023, pp. 328–348.

[8] M. Petersen and R. Tedrake, “Growing convex collision-free regions in configuration space using nonlinear programming,” *arXiv preprint arXiv:2303.14737*, 2023.

[9] T. Marcucci, J. Umenberger, P. Parrilo, and R. Tedrake, “Shortest paths in graphs of convex sets,” *SIAM Journal on Optimization*, vol. 34, no. 1, pp. 507–532, 2024.

[10] T. Cohn, M. Petersen, M. Simchowitz, and R. Tedrake, “Non-Euclidean Motion Planning with Graphs of Geodesically-Convex Sets,” in *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, Jul. 2023. DOI: 10.15607/RSS.2023.XIX.057.

[11] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, “Probabilistic roadmaps for path planning in high-dimensional configuration spaces,” *IEEE transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[12] S. M. LaValle, “Rapidly-exploring random trees: A new tool for path planning,” 1998.

[13] R. Tedrake, *Robotic Manipulation, Perception, Planning, and Control.* 2022. URL: http://manipulation.mit.edu.

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

[15] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal, "STOMP: Stochastic trajectory optimization for motion planning," in *2011 IEEE international conference on robotics and automation*, IEEE, 2011, pp. 4569–4574.

[16] M. Toussaint, "Newton methods for k-order Markov constrained motion problems," *arXiv preprint arXiv:1407.0414*, 2014.

[17] Gurobi Optimization, LLC, *Gurobi Optimizer Reference Manual*, 2023.

[18] M. ApS, *MOSEK Optimization Suite*, 2019.

[19] D. Ioan, I. Prodan, S. Olaru, F. Stoican, and S.-I. Niculescu, "Mixed-integer programming in motion planning," *Annual Reviews in Control*, vol. 51, pp. 65–87, 2021.

[20] B. Landry, R. Deits, P. R. Florence, and R. Tedrake, "Aggressive quadrotor flight through cluttered environments using mixed integer programming," in *2016 IEEE international conference on robotics and automation (ICRA)*, IEEE, 2016, pp. 1469–1475.

[21] R. Deits and R. Tedrake, "Footstep planning on uneven terrain with mixed-integer convex optimization," in *2014 IEEE-RAS international conference on humanoid robots*, IEEE, 2014, pp. 279–286.

[22] A. K. Valenzuela, "Mixed-integer convex optimization for planning aggressive motions of legged robots over rough terrain," Ph.D. dissertation, Massachusetts Institute of Technology, 2016.

[23] B. Aceituno-Cabezas, C. Mastalli, H. Dai, M. Focchi, A. Radulescu, D. G. Caldwell, J. Cappelletto, J. C. Grieco, G. Fernández-López, and C. Semini, "Simultaneous contact, gait, and motion planning for robust multilegged locomotion via mixed-integer convex optimization," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2531–2538, 2017.

[24] H. Dai, G. Izatt, and R. Tedrake, "Global inverse kinematics via mixed-integer convex optimization," *The International Journal of Robotics Research*, vol. 38, no. 12-13, pp. 1420–1441, 2019.

[25] A. Adu-Bredu, N. Devraj, and O. C. Jenkins, "Optimal Constrained Task Planning as Mixed Integer Programming," *arXiv preprint arXiv:2211.09632*, 2022.

[26] A. Tika, F. Gashi, and N. Bajcinca, "Robot Online Task and Trajectory Planning using Mixed-Integer Model Predictive Control," in *2022 European Control Conference (ECC)*, IEEE, 2022, pp. 2005–2011.

[27] S. Saha and A. A. Julius, "Task and motion planning for manipulator arms with metric temporal logic specifications," *IEEE robotics and automation letters*, vol. 3, no. 1, pp. 379–386, 2017.

[28] J.-s. Yi, T. A. Luong, H. Chae, *et al.*, "An Online Task-Planning Framework Using Mixed Integer Programming for Multiple Cooking Tasks Using a Dual-Arm Robot," *Applied Sciences*, vol. 12, no. 8, p. 4018, 2022.

[29] J. Chen, J. Li, Y. Huang, C. Garrett, D. Sun, C. Fan, A. Hofmann, C. Mueller, S. Koenig, and B. C. Williams, "Cooperative task and motion planning for multi-arm assembly systems," *arXiv preprint arXiv:2203.02475*, 2022.

[30] H. T. Suh, X. Xiong, A. Singletary, A. D. Ames, and J. W. Burdick, "Energy-efficient motion planning for multi-modal hybrid locomotion," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2020, pp. 7027–7033.

[31] N. D. Ratliff, J. Issac, D. Kappler, S. Birchfield, and D. Fox, "Riemannian motion policies," *arXiv preprint arXiv:1801.02854*, 2018.

[32] C.-A. Cheng, M. Mukadam, J. Issac, S. Birchfield, D. Fox, B. Boots, and N. Ratliff, "RMPflow: A computational graph for automatic motion policy generation," in *International Workshop on the Algorithmic Foundations of Robotics*, Springer, 2018, pp. 441–457.

[33] M. A. Rana, A. Li, D. Fox, S. Chernova, B. Boots, and N. Ratliff, "Towards Coordinated Robot Motions: End-to-End Learning of Motion Policies on Transform Trees," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2021, pp. 7792–7799.

[34] K. Van Wyk, M. Xie, A. Li, *et al.*, "Geometric fabrics: Generalizing classical mechanics to capture the physics of behavior," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 3202–3209, 2022.

[35] A. Bylard, R. Bonalli, and M. Pavone, "Composable geometric motion policies using multi-task pullback bundle dynamical systems," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2021, pp. 7464–7470.

[36] H. Klein, N. Jaquier, A. Meixner, and T. Asfour, "A Riemannian take on human motion analysis and retargeting," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2022, pp. 5210–5217.

[37] M. P. Do Carmo, *Riemannian geometry*. Springer, 1992, vol. 6.

[38] J. M. Lee, "Smooth manifolds," in *Introduction to smooth manifolds*, Springer, 2013.

[39] J. M. Lee, *Introduction to Riemannian manifolds*. Springer, 2018, vol. 176.

[40] N. Boumal, "An introduction to optimization on smooth manifolds," Jun. 2022. URL: https://www.nicolasboumal.net/book.

[41] S. B. Myers and N. E. Steenrod, "The group of isometries of a Riemannian manifold," *Annals of Mathematics*, pp. 400–416, 1939.

[42] M. Atcçeken and S. Keleş, "On the product Riemannian manifolds," *Differential Geometry-Dynamical Systems*, vol. 5, no. 1, pp. 1–7, 2003.

[43] N. K. Vishnoi, "Geodesic convex optimization: Differentiation on manifolds, geodesics, and convexity," *arXiv preprint arXiv:1806.06373*, 2018.

[44] H. Zhang and S. Sra, "First-order methods for geodesically convex optimization," in *Conference on Learning Theory*, PMLR, 2016, pp. 1617–1638.

[45] M. Bacak, *Convex Analysis and Optimization in Hadamard Spaces*. Walter de Gruyter GmbH & Co KG, 2014, vol. 22.

[46] H. Marchand, A. Martin, R. Weismantel, and L. Wolsey, "Cutting planes in integer and mixed integer programming," *Discrete Applied Mathematics*, vol. 123, no. 1-3, pp. 397–446, 2002.

[47] D. Burago, Y. Burago, and S. Ivanov, *A course in metric geometry*. American Mathematical Society, 2001, vol. 33.

[48] J. Väisälä, "A proof of the Mazur-Ulam theorem," *The American mathematical monthly*, vol. 110, no. 7, pp. 633–635, 2003.

[49] F. C. Park and B. Ravani, "Bézier curves on Riemannian manifolds and Lie groups with kinematics applications," *Journal of Mechanical Design*, vol. 117, no. 1, pp. 36–40, Mar. 1995.

[50] T. Popiel and L. Noakes, "Bézier curves and C^2 interpolation in Riemannian manifolds," *Journal of Approximation Theory*, vol. 148, no. 2, pp. 111–127, 2007.

[51] L. Jaillet and J. M. Porta, "Path planning under kinematic constraints by rapidly exploring manifolds," *IEEE Transactions on Robotics*, vol. 29, no. 1, pp. 105–117, 2013.

[52] C. Liverani, "Implicit function theorem 1(a quantitative version)," *retrieved January*, vol. 13, 2019.

[53] T. Cohn, S. Shaw, M. Simchowitz, and R. Tedrake, "Constrained bimanual planning with analytic inverse kinematics," in *2024 International Conference on Robotics and Automation (ICRA)*, IEEE, 2024, © IEEE 2024.

[54] A. V. Knyazev and P. Zhu, "Principal angles between subspaces and their tangents," *arXiv preprint arXiv:1209.0523*, 2012.

[55] P. Cignoni, C. Rocchini, and R. Scopigno, "Metro: measuring error on simplified surfaces," in *Computer graphics forum*, Wiley Online Library, vol. 17, 1998, pp. 167–174.

[56] J. Cheeger, D. G. Ebin, and D. G. Ebin, *Comparison theorems in Riemannian geometry*. North-Holland publishing company Amsterdam, 1975, vol. 9.

[57] R. Tedrake and the Drake Development Team, *Drake: Model-based design and verification for robotics*, 2019. URL: https://drake.mit.edu.

[58] P. E. Gill, W. Murray, and M. A. Saunders, "SNOPT: An SQP algorithm for large-scale constrained optimization," *SIAM Rev.*, vol. 47, pp. 99–131, 2005.

[59] P. E. Gill, W. Murray, M. A. Saunders, and E. Wong, "User's Guide for SNOPT 7.7: Software for Large-Scale Nonlinear Programming," Department of Mathematics, University of California, San Diego, La Jolla, CA, Center for Computational Mathematics Report CCoM 18-1, 2018.

[60] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004. URL: https://web.stanford.edu/~boyd/cvxbook/.

[61] E. B. Dam, M. Koch, and M. Lillholm, *Quaternions, interpolation and animation*. Citeseer, 1998, vol. 2.

[62] V. A. Dahl, A. B. Dahl, and R. Larsen, "Surface detection using round cut," in *2014 2nd International Conference on 3D Vision*, IEEE, vol. 2, 2014, pp. 82–89.

[63] L. Blumenson, "A derivation of n-dimensional spherical coordinates," *The American Mathematical Monthly*, vol. 67, no. 1, pp. 63–66, 1960.

[64] F. Krebs and T. Asfour, "A bimanual manipulation taxonomy," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 11 031–11 038, 2022.

[65] R. Holladay, T. Lozano-Pérez, and A. Rodriguez, "Robust planning for multi-stage forceful manipulation," *The International Journal of Robotics Research*, vol. 43, no. 3, pp. 330–353, 2024. DOI: 10.1177/02783649231198560. eprint: https://doi.org/10.1177/02783649231198560. URL: https://doi.org/10.1177/02783649231198560.

[66] D. Berenson, S. S. Srinivasa, D. Ferguson, and J. J. Kuffner, "Manipulation planning on constraint manifolds," in *2009 IEEE international conference on robotics and automation*, IEEE, 2009, pp. 625–632.

[67] B. Krauskopf, H. M. Osinga, and J. Galán-Vioque, *Numerical continuation methods for dynamical systems*. Springer, 2007, vol. 2.

[68] M. Bonilla, E. Farnioli, L. Pallottino, and A. Bicchi, "Sample-based motion planning for soft robot manipulators under task constraints," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2015, pp. 2522–2527.

[69] R. Bonalli, A. Cauligi, A. Bylard, T. Lew, and M. Pavone, "Trajectory optimization on manifolds: A theoretically-guaranteed embedded sequential convex programming approach," in *Proceedings of Robotics: Science and Systems*, Freiburgim Breisgau, Germany, Jun. 2019. DOI: 10.15607/RSS.2019.XV.078.

[70] M. Bonilla, L. Pallottino, and A. Bicchi, "Noninteracting constrained motion planning and control for robot manipulators," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2017, pp. 4038–4043.

[71] M. Stilman, "Global manipulation planning in robot joint space with task constraints," *IEEE Transactions on Robotics*, vol. 26, no. 3, pp. 576–584, 2010.

[72] J. Mirabel, S. Tonneau, P. Fernbach, A.-K. Seppälä, M. Campana, N. Mansard, and F. Lamiraux, "Hpp: A new software for constrained motion planning," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2016, pp. 383–389.

[73] L. Jaillet and J. M. Porta, "Path planning under kinematic constraints by rapidly exploring manifolds," *IEEE Transactions on Robotics*, vol. 29, no. 1, pp. 105–117, 2012.

[74] O. Bohigas, M. E. Henderson, L. Ros, M. Manubens, and J. M. Porta, "Planning singularity-free paths on closed-chain manipulators," *IEEE Transactions on Robotics*, vol. 29, no. 4, pp. 888–898, 2013.

[75] B. Kim, T. T. Um, C. Suh, and F. C. Park, "Tangent bundle rrt: A randomized algorithm for constrained motion planning," *Robotica*, vol. 34, no. 1, pp. 202–225, 2016.

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

[77] L. Han, L. Rudolph, J. Blumenthal, and I. Valodzin, "Convexly stratified deformation spaces and efficient path planning for planar closed chains with revolute joints," *The International Journal of Robotics Research*, vol. 27, no. 11-12, pp. 1189–1212, 2008.

[78] T. McMahon, S. Thomas, and N. M. Amato, "Sampling-based motion planning with reachable volumes for high-degree-of-freedom manipulators," *The International Journal of Robotics Research*, vol. 37, no. 7, pp. 779–817, 2018.

[79] F. Zha, Y. Liu, W. Guo, P. Wang, M. Li, X. Wang, and J. Li, "Learning the metric of task constraint manifolds for constrained motion planning," *Electronics*, vol. 7, no. 12, p. 395, 2018.

[80] I. A. Şucan and S. Chitta, "Motion planning with constraints using configuration space approximations," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2012, pp. 1904–1910.

[81] Z. Kingston, M. Moll, and L. E. Kavraki, "Sampling-based methods for motion planning with constraints," *Annual review of control, robotics, and autonomous systems*, vol. 1, pp. 159–185, 2018.

[82] R. Bordalba, T. Schoels, L. Ros, J. M. Porta, and M. Diehl, "Direct collocation methods for trajectory optimization in constrained robotic systems," *IEEE Transactions on Robotics*, 2022.

[83] L. Han and N. M. Amato, "A kinematics-based probabilistic roadmap method for closed chain systems," in *Algorithmic and computational robotics*, AK Peters/CRC Press, 2001, pp. 243–251.

[84] J. Cortés and T. Siméon, "Sampling-based motion planning under kinematic loop-closure constraints," in *Algorithmic Foundations of Robotics VI*, Springer, 2005, pp. 75–90.

[85] M. Gharbi, J. Cortés, and T. Simeon, "A sampling-based path planner for dual-arm manipulation," in *2008 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, IEEE, 2008, pp. 383–388.

[86] J. Wang, S. Liu, B. Zhang, and C. Yu, "Inverse kinematics-based motion planning for dual-arm robot with orientation constraints," *International Journal of Advanced Robotic Systems*, vol. 16, no. 2, p. 1 729 881 419 836 858, 2019.

[87] F. Kanehiro, E. Yoshida, and K. Yokoi, "Efficient reaching motion planning and execution for exploration by humanoid robots," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2012, pp. 1911–1916.

[88] D. Rakita, B. Mutlu, and M. Gleicher, "Relaxedik: Real-time synthesis of accurate and feasible robot arm motion.," in *Robotics: Science and Systems*, Pittsburgh, PA, vol. 14, 2018, pp. 26–30.

[89] F. Burget, A. Hornung, and M. Bennewitz, "Whole-body motion planning for manipulation of articulated objects," in *2013 IEEE International Conference on Robotics and Automation*, IEEE, 2013, pp. 1656–1662.

[90] B. Siciliano, O. Khatib, and T. Kröger, *Springer handbook of robotics*. Springer, 2008, vol. 200.

[91] R. Tedrake, *Robotic Manipulation, Perception, Planning, and Control*. 2023. URL: http://manipulation.mit.edu.

[92] M. Raghavan and B. Roth, "Inverse kinematics of the general 6r manipulator and related linkages," *Journal of Mechanical Design*, vol. 115, no. 3, pp. 502–508, Sep. 1993. DOI: 10.1115/1.2919218. URL: https://doi.org/10.1115/1.2919218.

[93] J. Nielsen and B. Roth, "On the kinematic analysis of robotic mechanisms," *The International Journal of Robotics Research*, vol. 18, no. 12, pp. 1147–1160, 1999.

[94] S. Xie, L. Sun, G. Chen, Z. Wang, and Z. Wang, "A novel solution to the inverse kinematics problem of general 7r robots," *IEEE Access*, vol. 10, pp. 67 451–67 469, 2022.

[95] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics*. Springer London, 2009. DOI: 10.1007/978-1-84628-642-1.

[96] R. Diankov, "Automated construction of robotic manipulation programs," Ph.D. dissertation, Carnegie Mellon University, The Robotics Institute Pittsburgh, 2010.

[97] R. Diankov and J. Kuffner, "Openrave: A planning architecture for autonomous robotics," *Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-08-34*, vol. 79, 2008.

[98] K. P. Hawkins, "Analytic inverse kinematics for the universal robots ur-5/ur-10 arms," *Georgia Institute of Technology, Tech. Rep*, 2013.

[99] K. Hauser, "Continuous pseudoinversion of a multivariate function: Application to global redundancy resolution," in *Algorithmic Foundations of Robotics XII: Proceedings of the Twelfth Workshop on the Algorithmic Foundations of Robotics*, Springer, 2020, pp. 496–511.

[100] A. Hemami, "A more general closed-form solution to the inverse kinematics of mechanical arms," *Advanced robotics*, vol. 2, no. 4, pp. 315–325, 1987.

[101] J. M. Hollerbach, "Optimum kinematic design for a seven degree of freedom manipulator," in *Robotics research: The second international symposium*, Citeseer, 1985, pp. 215–222.

[102] M. Shimizu, H. Kakuya, W.-K. Yoon, K. Kitagaki, and K. Kosuge, "Analytical inverse kinematic computation for 7-dof redundant manipulators with joint limits and its application to redundancy resolution," *IEEE Transactions on robotics*, vol. 24, no. 5, pp. 1131–1142, 2008.

[103] C. Faria, F. Ferreira, W. Erlhagen, S. Monteiro, and E. Bicho, "Position-based kinematics for 7-dof serial manipulators with global configuration control, joint limit and singularity avoidance," *Mechanism and Machine Theory*, vol. 121, pp. 317–334, 2018.

[104] Y. He and S. Liu, "Analytical inverse kinematics for franka emika panda–a geometrical solver for 7-dof manipulators with unconventional design," in *2021 9th International Conference on Control, Mechatronics and Automation (ICCMA)*, IEEE, 2021, pp. 194–199.

[105] G. K. Singh and J. Claassens, "An analytical solution for the inverse kinematics of a redundant 7dof manipulator with link offsets," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2010, pp. 2976–2982.

[106] J. J. Kuffner and S. M. LaValle, "Rrt-connect: An efficient approach to single-query path planning," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, IEEE, vol. 2, 2000, pp. 995–1001.

[107] R. Bohlin and L. E. Kavraki, "Path planning using lazy prm," in *Proceedings 2000 ICRA. Millennium conference. IEEE international conference on robotics and automation. Symposia proceedings (Cat. No. 00CH37065)*, IEEE, vol. 1, 2000, pp. 521–528.

[108] L. Jaillet, J. Cortés, and T. Siméon, "Sampling-based path planning on configuration-space costmaps," *IEEE Transactions on Robotics*, vol. 26, no. 4, pp. 635–646, 2010.

[109] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The international journal of robotics research*, vol. 30, no. 7, pp. 846–894, 2011.

[110] I. Ko, B. Kim, and F. C. Park, "Randomized path planning on vector fields," *The International Journal of Robotics Research*, vol. 33, no. 13, pp. 1664–1682, 2014.

[111] O. Salzman and D. Halperin, "Asymptotically near-optimal rrt for fast, high-quality motion planning," *IEEE Transactions on Robotics*, vol. 32, no. 3, pp. 473–483, 2016.

[112] M. Otte and E. Frazzoli, "Rrtx: Asymptotically optimal single-query sampling-based motion planning with quick replanning," *The International Journal of Robotics Research*, vol. 35, no. 7, pp. 797–822, 2016.

[113] M. Toussaint, "A tutorial on newton methods for constrained trajectory optimization and relations to slam, gaussian process smoothing, optimal control, and probabilistic inference," *Geometric and numerical foundations of movements*, pp. 361–392, 2017.

[114] E. Fernández González *et al.*, "Generative multi-robot task and motion planning over long horizons," Ph.D. dissertation, Massachusetts Institute of Technology, 2018.

[115] R. Deits and R. Tedrake, "Efficient mixed-integer planning for uavs in cluttered environments," in *2015 IEEE international conference on robotics and automation (ICRA)*, IEEE, 2015, pp. 42–49.

[116] D. H. Gottlieb, "Topology and the robot arm," *Acta Applicandae Mathematica*, vol. 11, pp. 117–121, 1988.

[117] J. W. Burdick, “On the inverse kinematics of redundant manipulators: Characterization of the self-motion manifolds,” in *Advanced Robotics: 1989: Proceedings of the 4th International Conference on Advanced Robotics Columbus, Ohio, June 13–15, 1989*, Springer, 1989, pp. 25–34.

[118] C. L. Luck and S. Lee, “Self-motion topology for redundant manipulators with joint limits,” in *[1993] Proceedings IEEE International Conference on Robotics and Automation*, IEEE, 1993, pp. 626–631.

[119] J. M. Lee and J. M. Lee, *Smooth manifolds*. Springer, 2012.

[120] R. Tedrake, *Underactuated Robotics, Algorithms for Walking, Running, Swimming, Flying, and Manipulation*. 2023. URL: https://underactuated.csail.mit.edu.

[121] F. Schwarzer, M. Saha, and J.-C. Latombe, “Exact collision checking of robot paths,” *Algorithmic foundations of robotics V*, pp. 25–41, 2004.

[122] I. A. Şucan, M. Moll, and L. E. Kavraki, “The Open Motion Planning Library,” *IEEE Robotics & Automation Magazine*, vol. 19, no. 4, pp. 72–82, Dec. 2012, https://ompl.kavrakilab.org. DOI: 10.1109/MRA.2012.2205651.

[123] N. Boumal, B. Mishra, P.-A. Absil, and R. Sepulchre, “Manopt, a Matlab toolbox for optimization on manifolds,” *Journal of Machine Learning Research*, vol. 15, no. 42, pp. 1455–1459, 2014. URL: https://www.manopt.org.

[124] C. Liu and N. Boumal, “Simple algorithms for optimization on riemannian manifolds with constraints,” *Applied Mathematics & Optimization*, vol. 82, no. 3, pp. 949–981, 2020.

[125] M. Weber and S. Sra, “Riemannian optimization via frank-wolfe methods,” *Mathematical Programming*, vol. 199, no. 1, pp. 525–556, 2023.

[126] W. Ballmann, “Manifolds of non positive curvature,” in *Arbeitstagung Bonn 1984: Proceedings of the meeting held by the Max-Planck-Institut für Mathematik, Bonn June 15–22, 1984*, Springer, 2006, pp. 261–268.

[127] C. Criscitiello and N. Boumal, “An accelerated first-order method for non-convex optimization on manifolds,” *arXiv preprint arXiv:2008.02252*, 2020.

[128] I. R. Manchester and J.-J. E. Slotine, “Control contraction metrics: Convex and intrinsic criteria for nonlinear feedback design,” *IEEE Transactions on Automatic Control*, vol. 62, no. 6, pp. 3046–3053, 2017.

[129] M. Nickel and D. Kiela, “Poincaré embeddings for learning hierarchical representations,” *Advances in neural information processing systems*, vol. 30, 2017.

[130] A. Morgan and A. Sommese, “Computing all solutions to polynomial systems using homotopy continuation,” *Applied Mathematics and Computation*, vol. 24, no. 2, pp. 115–138, 1987.

[131] J. C. Dunn, “Global and asymptotic convergence rate estimates for a class of projected gradient processes,” *SIAM Journal on Control and Optimization*, vol. 19, no. 3, pp. 368–400, 1981.

[132] P. G. Ciarlet, *Mathematical Elasticity Volume I: Three-Dimensional Elasticity* (Studies in mathematics and its applications), en. London, England: Elsevier Science, Mar. 1988.

[133] É. Cartan, *Geometry of Riemannian Spaces: Lie Groups; History, Frontiers and Applications Series*. Math Science Press, 1983, vol. 13.

[134] C. Phillips-Grafflin, *Common Robotics Utilities*, 2023.