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

Thomas Cohn*, Mark Petersen†, Max Simchowitz*, and Russ Tedrake*
*Computer Science and Artificial Intelligence Laboratory (CSAIL)
Massachusetts Institute of Technology, Cambridge, Massachusetts 02139–4309
Email: {tcohn,msimchow,russt}@mit.edu
†School of Engineering and Applied Sciences (SEAS)
Harvard University, Cambridge, Massachusetts 02138–2933
Email: markpetersen@g.harvard.edu

Abstract—Computing optimal, collision-free trajectories for high-dimensional systems is a challenging problem. Sampling-based planners struggle with the dimensionality, whereas trajectory optimizers may get stuck in local minima due to inherent nonconvexities in the optimization landscape. The use of mixed-integer programming to encapsulate these nonconvexities and find globally optimal trajectories has recently shown great promise, thanks in part to tight convex relaxations and efficient approximation strategies that greatly reduce runtimes. These approaches were previously limited to Euclidean configuration spaces, precluding their use with mobile bases or continuous revolute joints. In this paper, we handle such scenarios by modeling configuration spaces as Riemannian manifolds, and we describe a reduction procedure for the zero-curvature case to a mixed-integer convex optimization problem. We demonstrate our results on various robot platforms, including producing efficient collision-free trajectories for a PR2 bimanual mobile manipulator.

Fig. 1: The start and goal pose for one of our motion planning experiments, using the PR2 bimanual mobile manipulator.

I. INTRODUCTION
Planning the motion of robots through environments with obstacles is a long-standing and ever-present problem in robotics. In this paper, 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 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. Trajectory optimizers formulate the motion planning problem as an optimization problem. This problem is inherently non-convex when there are obstacles in the scene, so trajectory optimizers 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 at all.

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 et al. [1, 2] 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. Motion planning becomes a shortest-path problem in this space. This GCS-Planner has been successfully applied to challenging, high-dimensional problems, including bimanual manipulation problems.

However, GCS-Planner 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. However, this does preclude
us from considering nonconvex costs and constraints, such as
derived from nonlinear dynamics and closed kinematic
chains. We also cannot handle ball joints, as their configuration
spaces lead to unavoidable nonconvexities, which we describe
in Subsection VI-D.

In this paper, we formulate the general problem of shortest-
path motion planning around obstacles on Riemannian mani-
folds. 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 continu-
ous revolute joints and mobile bases. We show that in this
case, our theoretical developments lead to simple and elegant
modifications to the original GCS-Planner. We entitle this
generalization GGCS-Planner, and demonstrate its efficacy
with several challenging motion planning experiments.

II. RELATED WORK

In the world of continuous motion planning around ob-
cstacles, 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 founda-
tional sampling-based planning algorithms are Probabilistic
Roadmaps (PRMs) [2] and Rapidly-Exploring Random Trees
(RRTs) [3]. 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
[4 §7.2], CHOMP [5], STOMP [6], and KOMO [7]. Trajec-
tory 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 often cannot
achieve global optimality and can often fail to produce feasible
trajectories when solutions exist.

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 [9, 10] have been brought to bear. The survey paper
of Ioan et al. [11] provides an overview of the use of MIP
in motion planning. Besides the work of Marcucci et al. [11],
Landry et al. [12] used MIP to plan aggressive quadrotor
flights through obstacle-dense environments. MIP has been
used to plan footstep locations for humanoid robots [13] and
for quadrupeds [14, 15]. Dai et al. [16] used MIP to globally
solve the inverse kinematics problem. Finally, MIP has seen
extensive use in hybrid task and motion planning [17, 18, 19
20, 21].

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 [22, 2]. GCS
in particular uses an MIP formulation with a small number of
integer variables, making branch-and-bound tractable. Fur-
thermore, the convex relaxation is tight, enabling efficient
approximation by solving only a convex problem combined
with a randomized rounding strategy. [2] 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. Riemann-
nian Motion Policies (RMPs) [23] 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 [24, 25] and generalization [26, 27].
Klein et al. [28] envision Riemannian geometry as a tool for
generating and combining elegant motion synergies for
complex robotic systems.

III. PRELIMINARIES

In this section, we cover some of the relevant mathematical
background. We supply intuitive definitions; for further refer-
cence on Riemannian geometry, see the textbooks of Do Carmo
[29] and Lee [30, 31]. Boumal [32] provides an excellent
overview of optimization over manifolds. We use the notation
\[ n = \{1, \ldots, n\}. \]

A. 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
\[
\tau_{\mathcal{U},\mathcal{V}} = \psi_\mathcal{V}^{-1} \circ \psi_\mathcal{U} \mid_{\psi_\mathcal{U}(\mathcal{U} \cap \mathcal{V})}
\]
A collection of charts whose domains cover the manifold is
called an atlas. We only consider \( C^\infty \)-smooth manifolds, where
all transition maps in the atlas are \( C^\infty \).

For each \( p \in \mathcal{M} \), the tangent space \( T_p \mathcal{M} \) is a \( d\)-dimensional
vector space representing the set of directional derivatives at

\[
\tau_{\mathcal{U},\mathcal{V}} = \psi_\mathcal{V}^{-1} \circ \psi_\mathcal{U} \mid_{\psi_\mathcal{U}(\mathcal{U} \cap \mathcal{V})}
\]

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

For each \( p \in \mathcal{M} \), the tangent space \( T_p \mathcal{M} \) is a \( d\)-dimensional
vector space representing the set of directional derivatives at
Given a differentiable curve $\gamma : (-\epsilon, \epsilon) \to M$ with $p = \gamma(0)$, this affords an interpretation of the velocity of $\gamma$ at $p$, $\gamma'(0)$, as an element of $T_pM$. For a smooth map of manifolds $f : M \to N$, the pushforward of $f$ at $p$ is a linear map $f_*p : T_pM \to T_{f(p)}N$, generalizing the Jacobian matrix. The pushforward is defined so that, with $\gamma$ defined as above, $f_*p(\gamma'(0)) = (f \circ \gamma)'(0)$.

A Riemannian metric $g$ is a smoothly-varying positive-definite bilinear form over $M$ that gives each tangent space $T_pM$ an inner product $\langle \cdot, \cdot \rangle_p$. The pair $(M, g)$ is a Riemannian manifold, and we frequently refer to $M$ exclusively when the choice of metric is unambiguous. A Riemannian metric allows one to measure the length of a curve, invariant to reparametrizations.

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

$$d_M(p, q) = \inf \{ L(\gamma) | \gamma \in C^1_{pw}([0, 1], M), \gamma(0) = p, \gamma(1) = q \}$$

where $C^1_{pw}([0, 1], M)$ is the set of parametric piecewise-continuously differentiable curves from the interval $[0, 1]$ to $M$. A curve that achieves this infimum need not exist in general. We also define $d_U(p, q)$ for $p, q \in U \subseteq M$ to be the infimum of the length of paths whose image is contained in $U$.

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

$$\langle u, v \rangle_p = \langle f_*p(u), f_*p(v) \rangle_{f(p)}$$

$\forall p \in M, \forall u, v \in T_pM$. If $f$ is also a diffeomorphism, and $M$ and $N$ are connected, then $f$ preserves distances, and is an isometry of metric spaces. The converse is also true.

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 $p \in M$ and $v \in T_pM$ 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 $\exp_p : T_pM \to M$, 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 $T_pM$.

The Cartesian product of two Riemannian manifolds is itself a Riemannian manifold. The curvature of the component manifolds influences the curvature of the product. Importanty, the product of flat manifolds is flat.

In Section V, this implies that a robot with a mobile base and (potentially many) continuous revolute joints has a flat configuration space.

**B. Convex Analysis on Manifolds**

To define convexity on a Riemannian manifold $(M, g)$, 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 $U \subseteq M$ is strongly geodesically convex (or g-convex) if $\forall p, q \in U$, there is a unique length-minimizing geodesic connecting $p$ and $q$, and it is entirely contained in $U$. 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 do not provide these guarantees. See further discussion.

G-convex neighborhoods exist around every point. For any $p \in M$, there is a convexity radius $r_p > 0$, such that the open ball $B_r(p) = \{ \exp_p(q) \mid q \in T_pM, ||q|| < r \}$ 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 $f : M \to \mathbb{R}$ is said to be geodesically convex (g-convex) on $U \subseteq M$ if, for any geodesic $\gamma : [0, 1] \to U$, $(f \circ \gamma)$ is a convex function on $[0, 1]$. That is, $\forall t \in [0, 1], f(\gamma(t)) \leq (1 - t)f(\gamma(0)) + tf(\gamma(1))$.
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 \cite{37,36}. 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 \cite{38}) may not generalize to Riemannian manifolds.

IV. PROBLEM STATEMENT

We may now precisely state our kinematic planning problem in the language of Riemannian geometry developed thus far. Let $Q = (\mathcal{Q}, g)$ be the configuration space of a robot, realized as a connected Riemannian manifold. 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

$$\arg\min_{\gamma} L(\gamma)$$

s.t.

$$\gamma(0) = p$$

$$\gamma(1) = q$$

where $L$ is the Riemannian arclength, given in Eq. (2). In the following sections, we develop machinery to solve optimization problems of this form.

V. GRAPHS OF GEODESICALLY-CONVEX SETS

We now introduce a graph of geodesically convex sets (GGCS), a Riemannian optimization framework that, in Section VI, we show is general enough to encompass Problem (5). A GGCS is a directed graph $G = (V, E)$ with certain properties, designed as a generalization of ordinary (Euclidean) GCS from Marcucci et al. \cite{11} §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 $\ell^e_v : \mathcal{Y}_u \times \mathcal{Y}_v \rightarrow \mathbb{R}_{\geq 0} \cup \{\infty\}$, 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 paper, 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 $\pi$ 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 $y_k \in \mathcal{Y}_{v_k}$; then the length of this path is

$$\ell^e_v(y_\pi) = \sum_{k=1}^K \ell^e_{v_{k-1}, v_k}(y_{k-1}, y_k)$$

Let $\Pi$ 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

$$\min_{\pi \in \Pi} \min_{y_\pi \in \mathcal{Y}_\pi} \ell^e_\pi(y_\pi)$$

Solving Problem (7) 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 $\psi_v$, and define $\mathcal{X}_v = \psi_v(\mathcal{Y}_v)$. For each edge $e = (u, v) \in E$, we define the edge cost on $(x_u, x_v) \in \mathcal{X}_u \times \mathcal{X}_v$:

$$\ell^e_v(x_u, x_v) = \ell^e_\pi(\psi_u^{-1}(x_u), \psi_v^{-1}(x_v))$$

This construction is shown in Fig. 3. To apply the GCS machinery, we require that the sets $\mathcal{X}_v$ and edge costs $\ell^e_v(x_u, x_v)$ are convex. As we show in Subsection VI-D, this is hopeless for manifolds with positive curvature. Luckily, for flat manifolds, convexity can be ensured, as will be shown in Subsection VI-A.

Importantly, many robot configuration spaces can be realized as flat manifolds. SE(2) is flat, all 1-dimensional manifolds are flat \cite{31} p. 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, because decomposing a ball joint into 1-DoF joints distorts the underlying geometry.

VI. MOTION PLANNING WITH GGCS

We want to use GGCS to make motion plans on Riemannian manifolds by solving Problem (5). Thus, we must prove that the optimal value is achieved by some trajectory that is feasible for a GGCS problem. We use the initialization ROSC (Riemannian Open Subset Closure) to describe closures of
open subsets of Riemannian manifolds, notably $\mathcal{M}$. ROSCs are
topological manifolds-with-boundary, but the boundary may not be smooth; for
example, polytopic obstacles lead to corners on the boundary of $\mathcal{M}$. 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 $p,q \in \mathcal{M}$, there exists a continuous curve $\gamma$ connecting them such that $L(\gamma) = d(p,q)$.

**Proof:** The proof follows by verifying that $\mathcal{M}$ satisfies the preconditions of Theorem 2.5.23 of Burago et al. [39, p. 50]. We defer the details to Appendix A.1.

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

These requirements will not hold in general, but we will discuss how to construct such an atlas in Subsection VI-B.

We can also extend each $\psi_v$ to be defined on $\overline{\mathcal{V}}$. With this information, we can prove a strong result about the shortest paths in $\mathcal{M}$.

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

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

**Proof:** Let $\gamma_0$ be a continuous minimizing path connecting $p$ to $q$ (guaranteed to exist by Theorem 1); we will use this to construct an appropriate $\gamma^*$. 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 $\gamma_{i-1}$ does not pass through $\overline{\mathcal{V}}_{v_i}$, let $\gamma_i = \gamma_{i-1}$. Otherwise, let $T_i = \{ t \mid \gamma_{i-1}(t) \notin \overline{\mathcal{V}}_{v_i} \}$, let $t_i = \min(T_i)$, and let $b'_i = \max(T_i)$. Then by the g-convexity of $\overline{\mathcal{V}}_{v_i}$, there is a unique minimizing geodesic $\alpha_i : [a'_i, b'_i] \to \overline{\mathcal{V}}_{v_i}$ connecting $\gamma_{i-1}(a'_i)$ and $\gamma_{i-1}(b'_i)$. Let $\tilde{\gamma}$ be a new curve, defined by

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

(9)

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

### A. 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.** $Q$ is flat. Also, each $\psi_v$ 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:

- $X_v = \psi_v(\overline{\mathcal{V}}_v)$ is convex.
- $\forall y_0, y_1 \in \overline{\mathcal{V}}_v, d(y_0, y_1) = ||\psi_v(y_0) - \psi_v(y_1)||_2$
- $\tau_{u,v}$ is a Euclidean isometry (see Lemma 2 in Appendix A.2), and hence affine 40.

The first two results are true because $\mathcal{V}_v$ is g-convex, $\psi_v$ maps geodesics to geodesics [31, p. 125], and geodesics are unique in Euclidean space. For most robotic configuration spaces we consider, $Q$ 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 [2], where decision variables describe line segments contained within each convex set. In particular, $\forall v \in V$, we have $x_v = (x_{v,0}, x_{v,1}) \in \mathbb{R}^2$, 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 $e$ $\ell_e(x_u, x_v) = d(\psi_u^{-1}(x_{u,0}), \psi_u^{-1}(x_{u,1})) = ||x_{u,0} - x_{u,1}||_2$

(10)

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

$$\psi_u^{-1}(x_{u,1}) = \psi_v^{-1}(x_{v,0}) \iff \tau_{u,v}(x_{u,1}) = x_{v,0}$$

(11)

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 [2]. If we have $p \in X_0$ and $q \in X_K$, then after solving the GCS problem, we obtain a path

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

(12)

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

$$y_\pi = (y_0, y_1, y_2, \ldots, y_K, y_{K+1} = q)$$

(13)

$$= (\psi_0^{-1}(x_{0,0}), \psi_1^{-1}(x_{1,0}), \ldots, \psi_K^{-1}(x_{K,0}), \psi_K^{-1}(x_{K,1}))$$

where we have removed duplicate points from the trajectory. This process is visualized for a simple cylinder manifold in Fig. 4.

For each $i \in \{0, \ldots, K\}$, $y_i$ and $y_{i+1}$ are contained in a g-convex set $\overline{\mathcal{V}}_i$, so there is a unique minimizing geodesic $\gamma_i$ connecting them and completely contained in $\overline{\mathcal{V}}_i$. Thus, a path
Thus, the optimal value of Problem (5) is no worse than the optimal value of the GCS problem.

Proof: Any feasible path \( x_\pi \) for the GCS problem yields a piecewise continuously differentiable curve \( \gamma_\pi \) whose image is contained in \( \widehat{\mathcal{M}} \) and connecting \( p \) to \( q \). Then the length of this curve satisfies

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

Thus, the optimal value of Problem (5) is no worse than the optimal value of the GCS problem.

Now, consider an optimal \( \gamma^* \) for Problem (5), with the properties of Theorem 2. Then \( \gamma^* \) is the concatenation of geodesics \( \gamma_1, \ldots, \gamma_K \), where \( \gamma_i : [0, 1] \to \mathcal{Y}_{v_i} \) for \( i = 1, \ldots, K \), and each \( v_i \) is distinct. Define \( x_\pi \) 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 \( x_\pi \) is feasible for the GCS problem. Thus, the GCS problem achieves the optimal value of Problem (5).

B. Construction of the Atlas

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

As was done in [2], we construct an inner approximation of C-Free using the extension of IRIS [41] 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. (11).

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 \( i \)th joint's configuration space, we constrain that joint to take values within an open ball of radius \( r_i \), centered at the seed point. (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 \( \mathcal{Q} = Q_1 \times \cdots \times Q_m \), where each \( Q_i \) has a convexity radius \( r_i \). Let \( (\mathcal{Y}, \psi) \) be a coordinate chart, with \( \psi \) a local isometry and \( \psi(\mathcal{Y}) \) convex in Euclidean space. If \( \mathcal{Q} \) is flat and the diameter of \( \text{proj}_{A_1(\mathcal{Y})} \) is less than \( 2r_i \), then \( \mathcal{Y} \) is g-convex.

We also assumed full coverage of \( \widehat{\mathcal{M}} \) 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.

C. More General Motion Planning

Marcucci et al. [2] extended GCS-Planner 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 [42, 43]. 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 \( \eta \) times for an edge \((i, j)\), with transition map \( \tau_{i,j} \). Let \( \gamma_i \) and \( \gamma_j \) be adjoining Bézier curve segments in \( \mathcal{Y}_i \) and \( \mathcal{Y}_j \), and let their \( k \)th 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

\[
(\tau_{i,j})_* \psi_{i,j}^{-1}(w) \nu_i^{(k)} = \nu_j^{(k)} \quad \forall l \in [\eta]
\]

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 VI-A), 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. (14) is a convex constraint.

D. Positive Curvature Induces Nonconvexity

At this point, we have shown that the flatness of \( \mathcal{Q} \) is sufficient for our formulation’s validity. It is natural to ask how essential this is, especially since SO(3), a manifold of great interest in robotics, has positive curvature. 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 SE(3) \( \cong \) SO(3) \( \times \) \( \mathbb{R}^3 \). 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.3).

Theorem 5. Let \( \mathcal{M} \) be a Riemannian manifold, let \( A_1 \in \mathcal{M} \) and \( u, v \in T_{A_1} \mathcal{M} \) such that \( K_l(u, v) > 0 \). Then for any neighborhood \( \mathcal{U} \) containing \( A_1 \), \( d : \mathcal{M}^2 \to \mathbb{R} \) is nonconvex on \( \mathcal{U}^2 \).

VII. Experiments

We demonstrate our GGCS planner 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. 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 \( \mathbb{R} \).

A. 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. 5. We also show an example of an optimal trajectory produced by GGCS-Planner, which “wraps around” the toroidal world. This plan was computed in 0.79 seconds.

B. 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 GGCS-Planner in Fig. 6, 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.

C. Modified KUKA iiwa Arm

We also demonstrate that GGCS-Planner 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 \( \mathbb{T}^1 \times \mathbb{T}^6 \). 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 GGCS-Planner 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
Fig. 6: Two plans produced by GGCS-Planner 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).

The experiment consisted of 14 motions, which were each planned individually, and we use the region refinement method from [45] 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).

D. 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 \( SE(2) \times T^2 \times I^{10} \cong T^3 \times T^{12} \). 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 GGCS-Planner 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. 9 we show several robot configurations along a trajectory produced by GGCS-Planner.

For the planning scenarios considered with the PR2, we compare GGCS-Planner to existing approaches. Trajectory lengths are listed in Table I and interactive recordings of all trajectories are available online at our results website. We compare our algorithm to kinematic trajectory optimization [5, §7.2] (abbreviated as Drake-Trajopt), utilizing the general nonlinear program solver SNOPT [46, 47]. Drake-Trajopt 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 [48, p. 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 GGCS-Planner, 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 GGCS-Planner, at the expense of colliding slightly with the environment (an example is shown in Fig. 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 GGCS-Planner 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 in Fig. 8 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° ≡ 360°). 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 GGCS and both GCS workarounds.

An interesting result is that the best case for the GCS workarounds is sometimes slightly better than GGCS. 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 GGCS, then, in the best case, their performance is indistinguishable.

VIII. DISCUSSION

In this paper, 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 GCS-Planner, 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 GGCS-Planner 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. In future work, we intend to generalize the techniques outlined in this paper for use in bimanual manipulation (with implicit kinematic constraints) and planning through contact. Although the planning and optimization frameworks used in GGCS-Planner 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.

<table>
<thead>
<tr>
<th>Experiment</th>
<th>GGCS-Planner</th>
<th>Drake-Trajopt</th>
<th>PRM</th>
<th>PRM + Drake-Trajopt</th>
</tr>
</thead>
<tbody>
<tr>
<td>1T to 1B</td>
<td>1.829</td>
<td>1.803</td>
<td>4.359</td>
<td>1.808</td>
</tr>
<tr>
<td>1CL to 1CR</td>
<td>2.255</td>
<td>2.204</td>
<td>9.219</td>
<td>2.182</td>
</tr>
<tr>
<td>1M to 4M</td>
<td>2.169 t = 0.272</td>
<td>14.554</td>
<td>5.874</td>
<td>6.482 / 10.478</td>
</tr>
<tr>
<td>1CL to 2CR</td>
<td>4.473</td>
<td>3.065 t = 2.155</td>
<td>12.110</td>
<td>4.406 t = 0.49</td>
</tr>
<tr>
<td>1CL to 3CR</td>
<td>8.182</td>
<td>10.263</td>
<td>15.250</td>
<td>7.106 t = 0.27, 2.02, 2.77</td>
</tr>
<tr>
<td>1CL to 4CR</td>
<td>4.382</td>
<td>7.583</td>
<td>17.459</td>
<td>6.005 t = 0.27, 0.355, 4.39</td>
</tr>
<tr>
<td>1T to 4B</td>
<td>4.538</td>
<td>8.781</td>
<td>12.351</td>
<td>5.945 t = 0.34, 0.68</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Experiment</th>
<th>GGCS-Planner</th>
<th>Drake-Trajopt</th>
<th>PRM</th>
<th>PRM + Drake-Trajopt</th>
</tr>
</thead>
<tbody>
<tr>
<td>1T to 1B</td>
<td>25.51</td>
<td>12.69</td>
<td>0.49</td>
<td>16.61</td>
</tr>
<tr>
<td>1CL to 1CR</td>
<td>39.42</td>
<td>15.23</td>
<td>0.49</td>
<td>16.11</td>
</tr>
<tr>
<td>1M to 4M</td>
<td>46.61</td>
<td>2.26</td>
<td>0.53</td>
<td>25.51</td>
</tr>
<tr>
<td>1CL to 2CR</td>
<td>62.87</td>
<td>9.74</td>
<td>0.54</td>
<td>21.48</td>
</tr>
<tr>
<td>1CL to 3CR</td>
<td>58.60</td>
<td>7.82</td>
<td>0.52</td>
<td>27.30</td>
</tr>
<tr>
<td>1CL to 4CR</td>
<td>66.15</td>
<td>4.32</td>
<td>0.54</td>
<td>40.10</td>
</tr>
<tr>
<td>1T to 4B</td>
<td>29.89</td>
<td>10.92</td>
<td>0.54</td>
<td>15.36</td>
</tr>
</tbody>
</table>

TABLE I: 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 GCS-Planner 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.

TABLE II: 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 GGCS-Planner runtimes.) Experiment names match Table I.

ACKNOWLEDGMENTS

We would like to thank Tobia Marcucci and Seiji Shaw for their valuable suggestions throughout the course of this work. Thank you to David von Wrangel for his assistance with the implementation of the PRM comparisons. This work was supported by (in alphabetical order) Amazon.com, PO No. 2D-06310236 and the Frederick and Barbara Cronin Fellowship.

REFERENCES


[33] Nisheeth K Vishnoi. Geodesic convex optimization:
Differentiation on manifolds, geodesics, and convexity.


[38] Hugues Marchand, Alexander Martin, Robert Weismantel, and Laurence Wolsey. Cutting planes in integer and mixed integer programming.


URL https://drake.mit.edu


Center for Computational Mathematics Report CCoM 18-1, Department of Mathematics, University of California, San Diego, La Jolla, CA, 2018.


2023.
APPENDIX A
PROOFS

A.1 Proof of Theorem 7

Lemma 1. 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 \( \gamma \), with \( p \in U_1, q \in U_n \), and \( U_1 \cap U_{i+1} \cap \overline{\mathcal{M}} \neq \emptyset \) for each \( i \). (Such a finite covering exists because the image of \( \gamma \) 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 \( i = 1, \ldots, n-1 \), \( \gamma_i \) is 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 \( \gamma_0 \) 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 \( \gamma_n \) 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 \( q \).

Fig. 11: The construction of two Levi-Civita parallelogramoids used in the proof of Theorem 5. \( d(A_0, B_0) < d(A_1, B_1) \) and \( d(A_2, B_2) < d(A_1, B_1) \), which demonstrates the nonconvexity of \( d \) around \( A_1 \). \( \angle A_0B_0B_1 \) and \( \angle A_2B_2B_1 \) are obtuse.

A.2 Proof of Theorem 2

Lemma 2. Let \( \langle Y_1, \psi_1 \rangle \) and \( \langle Y_2, \psi_2 \rangle \) be coordinate charts of \( \mathcal{M} \), with \( \psi_1 \) and \( \psi_2 \) local isometries, and \( Y_1 \) and \( Y_2 \) g-convex. Then there is a Euclidean isometry \( \xi \) such that \( \forall p \in Y_1 \cap Y_2, \psi_1(p) = (\xi \circ \psi_2)(p) \).

Proof: \( Y_1 \cap Y_2 \) 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 \( (\psi_1 \circ \psi_2^{-1})_{\cdot, p} \) is an orthogonal matrix for any \( p \). Thus, we can apply Theorem 1.8-1 of Ciarlet [39] p. 44).

A.3 Proof of Theorem 5

Proof of Theorem 4: For each \( i \in [m] \), we can construct a ball \( B_{s_i}(c_i) \supset \text{proj}_A(Y) \), with \( s_i < r_i \). Define \( Z = \bigcap_{i \in [m]} B_{s_i}(c_i) \), a g-convex set. Consider the Riemannian normal coordinates of \( \mathcal{Q} \) at \( c_1, \ldots, c_m \). This coordinate system, restricted to \( Z \), induces a coordinate chart \( \varphi \). Because \( \mathcal{Q} \) is flat, \( \varphi \) is a local isometry, so by Lemma 2 there is a Euclidean isometry \( \xi \) such that \( \varphi(Y) = \xi(\psi(Y)) \), so \( \varphi(Y) \) is convex. Thus, by Lemma 3 \( Y \) is g-convex.

Proof of Theorem 5: 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 \( \gamma \) on \( \mathcal{U}^2 \) such that \( d \circ \gamma \) achieves smaller values on its endpoints than at its center. This relies on the properties of specially constructed Levi-Civita parallelogramoids on \( \mathcal{U} \).

Fig. 11: The construction of two Levi-Civita parallelogramoids used in the proof of Theorem 5. \( d(A_0, B_0) < d(A_1, B_1) \) and \( d(A_2, B_2) < d(A_1, B_1) \), which demonstrates the nonconvexity of \( d \) around \( A_1 \). \( \angle A_0B_0B_1 \) and \( \angle A_2B_2B_1 \) are obtuse.
APPENDIX 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 VII-B 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 GGCS-Planner 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 paper. 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 VII-C 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.

The GGCS-Planner 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 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 GGCS-Planner, 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 GGCS-Planner: 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 VII-C are utilized). The KinematicTrajectoryOptimization class 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 [2], we use the PRM planner from the Common Robotics Utilities library [51], with the modifications described in Subsection VII-D. 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.